VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

DETEKCE AKTUÁLNÍHO PODLAŽÍ PŘI JÍZDĚ VÝTAHEM

FLOOR DETECTION DURING ELEVATOR RIDE

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Martin Havelka

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Jiří Krejsa, Ph.D.

SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadaní diplomové práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Bc. Martin Havelka**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Mechatronika
Vedoucí práce:	**doc. Ing. Jiří Krejsa, Ph.D.**
Akademický rok:	2020/21

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Detekce aktuálního podlaží při jízdě výtahem

Stručná charakteristika problematiky úkolu:

Jednou z netriviálních úloh v mobilní robotice je schopnost mobilního robotu pohybovat se mezi patry budov pomocí běžného osobního výtahu. Ke splnění této úlohy musí být robot schopen určit, v jakém patře se aktuálně nachází. Podstatou diplomové práce je detekce aktuálního podlaží pomocí fúze dostupných senzorických dat (palubního akcelerometru a obrazu palubní kamery robotu).

Cíle diplomové práce:

1. Seznamte se se způsoby fúze dat různého typu.
2. Vytvořte dostatečně mohutnou množinu obrazů informačních prvků výtahu.
3. Pomocí akcelerometrického datalogeru naměřte průběhy zrychlení při jízdě výtahem.
4. Implementujte vybranou metodu fůze akcelerometrických a obrazových dat.
5. Metodu ověřte a vyhodnoťte.

Seznam doporučené literatury:

MOHINDER S. GREWAL, A. P. ANDREWS: Kalman Filtering: Theory and Practice with MATLAB, Wiley-IEEE Press; 4th Edition, 2014.

FOURATI H.: Multisensor Data Fusion: From Algorithms and Architectural Design to Applications, CRC Press, 2015.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2020/21

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

Abstrakt

Tato diplomová práce se zabývá detekcí aktuálního podlaží při jízdě výtahem. Jedná se o funkcionalitu, která je nezbytná při pohybu robota ve vícepodlažní budově. Pro tuto úlohu je využita fúze akcelerometrických dat v průběhu jízdy výtahem a obrazových dat získaných z informačního displeje uvnitř výtahové kabiny. V rešeršní části jsou popsána již realizovaná řešení, metody datové fúze a možnosti klasifikace obrazu. Na základě této části byly vybrány vhodné přístupy pro řešení úlohy. V první fázi byly získány datové sady z různých typů výtahových kabin. Následně byl vyvinut algoritmus pro práci s daty z akcelerometrického senzoru. Byla vybrána a natrénována konvoluční neuronová síť, která byla využita pro klasifikaci obrazových dat. Následně byla implementována metoda datové fúze. Jednotlivé části byly samostatně otestovány a vyhodnoceny. Na základě jejich vyhodnocení byla provedena integrace do jednoho funkční systému, který byl úspěšně ověřen a otestován. Výsledná úspěšnost detekce při jízdě různými výtahy byla stanovena na 97%.

Klíčová slova

Detekce podlaží, Fúze senzorických dat, Bayesův filtr, Výtah, CNN, EfficientNet

Abstract

This diploma thesis deals with the detection of the current floor during elevator ride. This functionality is necessary for robot to move in multi-floor building. For this task, a fusion of accelerometric data during the ride of the elevator and image data obtained from the information display inside the elevator cabin is used. The research describes the already implemented solutions, data fusion methods and image classification options. Based on this part, suitable approaches for solving the problem were proposed. First, datasets from different types of elevator cabins were obtained. An algorithm for working with data from the accelerometric sensor was developed. A convolutional neural network, which was used to classify image data from displays, was selected and trained. Subsequently, the data fusion method was implemented. The individual parts were tested and evaluated. Based on their evaluation, integration into one functional system was performed. System was successfully verified and tested. Result of detection during the ride in different elevators was 97%.

Keywords

Floor detection, Multisensor data fusion, Bayes filter, Elevator, CNN, EfficientNet

Bibliografická citace:

HAVELKA, Martin. Detekce aktuálního podlaží při jízdě výtahem. Brno, 2021. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/132985. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav mechaniky těles, mechatroniky a biomechaniky. Vedoucí práce doc. Ing. Jiří Krejsa, Ph.D.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Detekce aktuálního podlaží při jízdě výtahem jsem vypracoval samostatně pod vedením vedoucího a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce.“

V Brně dne: **10. května 2021** Bc. Martin Havelka

Poděkování

Děkuji vedoucímu doc. Ing. Jiřímu Krejsovi, Ph.D. za odborné vedení a přátelský přístup při zpracování této diplomové práce. Dále děkuji kolegovi Martinu Černilovi za spolupráci a cenné rady. Na závěr chci poděkovat své rodině a manželce za podporu během celého studia.

V Brně dne: **10. května 2021** Bc. Martin Havelka

Obsah

1 ÚVOD

Jednou z netriviálních úloh v mobilní robotice je schopnost mobilního robotu pohybovat se mezi patry budov pomocí běžného osobního výtahu. Vyřešením této úlohy je možné rozšířit možnosti použití robota a tím tak zvýšit jeho efektivitu a rozsah úkolů, které dokáže splnit.

Mezi autonomní systémy pohybující se vícepodlažním prostorem, které byly vyvinuty na různých zahraničních univerzitách patří např. GRACE [1], MOPS [2] nebo The Fujitsu service robot [3]. Na základě realizovaných řešení je možné zhodnotit možné přístupy. Stále se však jedná o oblast, která nabízí široké možnosti při návrhu nových řešení. Důležitým požadavkem na systém je univerzálnost. Většina dosavadních řešení se zabývá pohybem v jasně definované budově a pomocí konkrétního osobního výtahu. Jedním z požadavků na vyvíjený systém je univerzální funkcionalita, která není závislá na konkrétním výtahu či budově. Tato specifikace vnáší do celé problematiky vyšší nároky.

Ke zvládnutí této úlohy je třeba, aby robot disponoval několika důležitými funkcemi, které jízdu výtahem umožní. Z všeobecného hlediska se jedná především o orientaci v mapě a navigaci, dále o schopnost detekovat ovládací prvky výtahu, interakce s těmito prvky (např. stlačení ovládacího prvku), o detekci aktuálního podlaží při jízdě výtahem a detekci otevření a zavření dveří. Nároky na takový systém jsou rychlost, přesnost a robustnost.

Robot musí disponovat několika základními funkcemi, které jsou pro možnost pohybu ve výtahu stěžejní. Cílem, který sjednocuje jednotlivé části této úlohy je vytvořit komplexní systém pro robotickou platformu Breach, který umožní jeho využití v budovách s osobním výtahem. Rozsah úlohy však neumožňuje zpracovat všechny její části v jedné práci. Proto byla rozdělena na jednotlivé funkční celky, které nejsou na sobě přímo závislé.

Tato práce se proto zabývá detekcí aktuálního podlaží při jízdě výtahem, která má být dosažena využitím fúze senzorických dat. Předpokládané řešení spočívá ve využití obrazových a akcelerometrických dat. Při vývoji je nutné uvažovat tak, aby byla zajištěna univerzálnost a robustnost celého systému. Z těchto požadavků je patrné, že pro tuto úlohu budou stěžejní senzorická data získaná z měření v různých výtahových kabinách. Vytvoření datových množin je pro tuto úlohu stejně důležité, jako systém samotný. Pomocí nich je možné vyvinutý systém účinně testovat. V řešení se dále počítá s využitím neuronových sítí, které pro fázi učení vyžadují velké množství dat.

V kapitole 2 je popsána rešerše, která má za cíl prozkoumat jednotlivé možnosti. S ohledem na požadavky a s ohledem na výstupy z rešeršní části budou v kapitole 3 jednotlivé cíle blíže specifikované a budou vybrány vhodné metody pro řešení úlohy detekce podlaží. Kapitola 4 popisuje vytvoření datových sad. Kapitola 5 má za cíl vysvětlit a popsat vyvinuté řešení. Kompletní algoritmus a implementace je popsána v kapitole 6. Na závěr je nutné celý systém otestovat, aby byla zhodnocena úspěšnost navrženého řešení.

2 REŠERŠNÍ ČÁST

Rešeršní část je členěna do třech hlavních témat. Nejprve jsou popsaná vybraná realizovaná řešení, která umožňují získat informace o problematice detekce aktuálního podlaží výtahu. Následně jsou zkoumány metody fúze senzorických dat. Při řešení této úlohy se předpokládá využití obrazových a akcelerometrických dat ze senzorů, proto jsou na závěr zmíněny možné přístupy při práci a zpracování dat. Ze všech zmíněných přístupů budou následně vybrány vhodné metody, které budou využity při realizaci řešení.

2.1 Realizovaná řešení a použité přístupy při detekci pater

Problematiku pohybu robota v osobním výtahu řeší několik výzkumů, které mají odlišné přístupy k řešení. Z tohoto důvodu je vhodné se nejprve seznámit s těmito výzkumy pro získání lepšího vhledu do problematiky.

2.1.1 Systém mezipodlažní navigace pro servisní robot

Jedná se o výzkum vědeckého týmu National Normal University na Taiwanu (reference [2]). Práce představuje systém mezipodlažní navigace pro servisní robot. Robot využívá prostředí ROS. Systém zahrnuje mapování, lokalizaci, plánování trasy robota a určení aktuálního patra výtahu. Úlohu rozpoznání aktuálního patra je řešena využitím konvoluční neuronová síť (CNN), konkrétně byla použita síť VGG16. Síť je natrénována tak, že při pohledu z výtahu při otevření dveří je schopna rozpoznat konkrétní podlaží. Test přesnosti detekce patra byl v rozsahu 93 % a 97 % dle patra.

Z toho vyplývá, že metoda není univerzální a funguje pouze pro daný výtah a pro patra, na která byla CNN natrénována.

2.1.2 Metoda jízdy výtahem pro mobilní robot H20

Jedná se o práci týmu z University of Rostock v Německu a University of Mosul v Iráku (reference [3]). Cílem bylo vyvinou řešení, které by umožnilo mobilnímu robotu H20 pohyb mezi patry pomocí výtahu. Toto řešení zahrnuje tři hlavní aspekty při autonomní jízdě výtahem. Jedná se o detekci přivolávacího tlačítka, detekce vnitřních tlačítek a čtení palubního displeje, který dává informaci o aktuálním patře. Byly využity různé techniky pro úpravu vstupních obrazových dat. K získání informace o pozici panelu byly využity markery, které vytyčují oblast zájmu (ROI).

Displej byl nalezen za pomoci adaptivní prahové hodnoty a dále za pomoci vyhledání objektů v obrázku. Dle znalosti velikosti displeje byl vybrán objekt, který této velikosti odpovídá a tím bylo nalezeno ROI pro displej. Ke čtení byl použit OCR program. Výsledky úspěšnosti detekce patra při počtu testů 100 byl dle zdroje 99 a 100 %.

2.1.3 Rozpoznání aktuálního podlaží

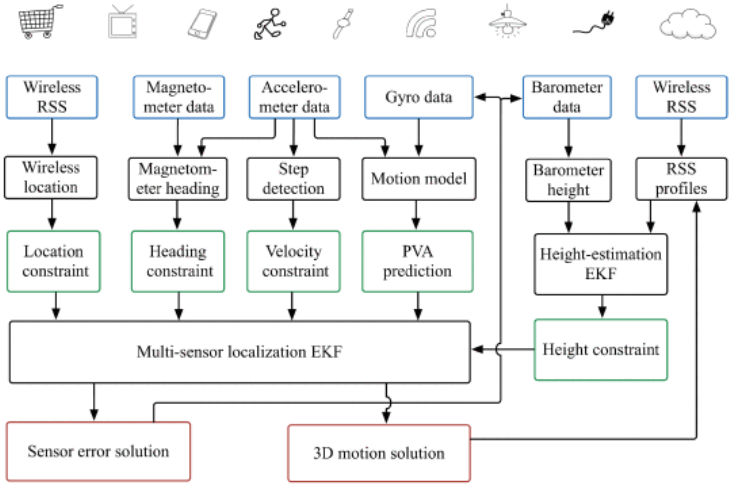
Jedná se o práci týmu University of Mlaya v Malaisii a Skkur University v Pakistanu (reference [4]). Práce se zabývá rozpoznáváním tlačítek a čísel aktuálního patra výtahu. Nejprve je

vytvořena datová sada čísel z displeje. Využita je extrakce červené barvy, z důvodu červených čísel na displeji. Vytvořená datová sada je použita jako vstup do feature extraktoru. Jsou použity metody histogram of oriented gradients (HOG) a bag-of-words (BoW). Jedná se o hybridní klasifikaci obrazu. Nejprve jsou nalezeny zajímavé prvky (features). Tyto prvky jsou použity jako vstup do umělé neuronové sítě (ANN). Výstupem z neuronové sítě je číslo podlaží.

Toto řešení má 99 % úspěšnost. Je nutné opět podotknout, že řešení je funkční pro tento konkrétní výtah.

2.1.4 3D lokalizace s robustní metodou detekce podlaží

Jedná se o výzkum týmu z univerzit v Calgary, Číně a Německu, který se zabývá 3D lokalizací za použití více senzorů (reference [5]). Je založen na RSS[1] lokalizaci pater. Výsledky z RSS detekce patra jsou spojeny s výsledky z barometrického senzoru a z inerciální jednotky. Dochází tak k fúzi dat. Fúzi zajišťuje rozšířený Kalmanův filtr (EKF), tím je zajištěno robustní řešení.



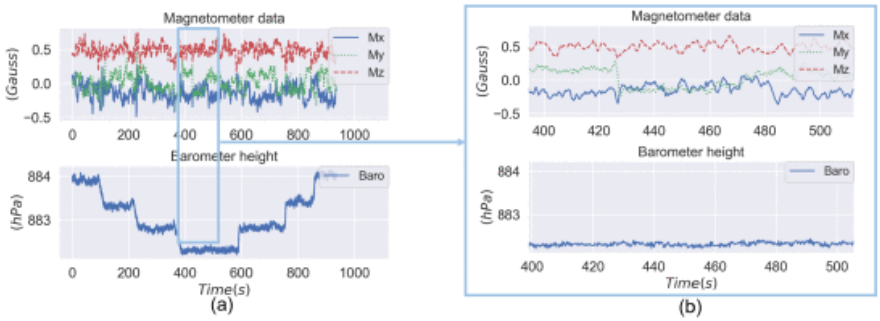
Obr. 2.1 – Ukázka schématu fúze dat

Na Obr. 2.1 (převzato z [5]) můžeme vidět diagram, kde jsou uvedena všechna použitá vstupní data a jejich další fúze. K určení výšky je využita bezdrátová RSS lokalizace spojená s daty z barometru, pomoci EKF je poté získána informace o výšce, v které se robot nachází. Druhý EKF z informací o pozici, směru, rychlosti a jejich predikci (PVA) v kombinaci s výškou, která je výstupem prvního EKF určí 3D lokalizaci a chybu senzorů.

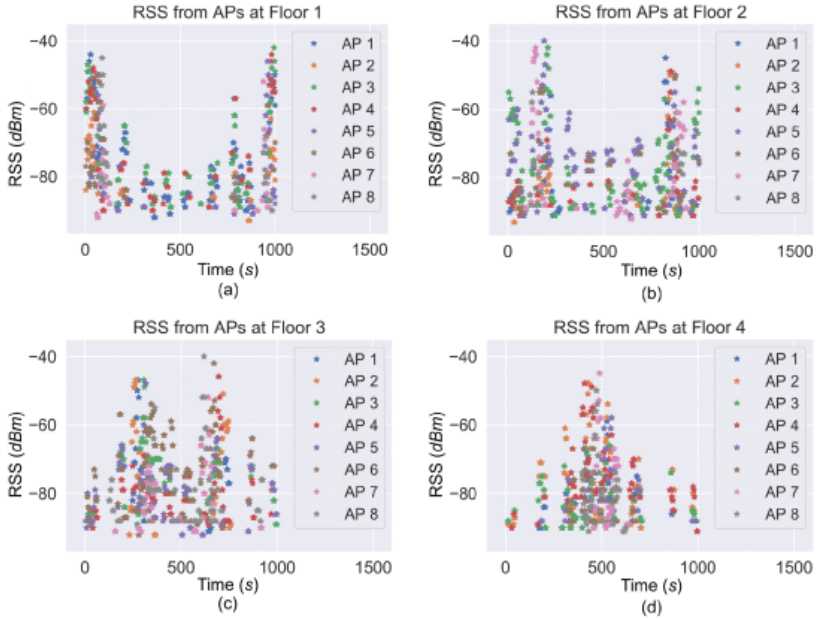
Z této práce nás nejvíce zajímá právě detekce pater. Vstupními daty jsou data z barometru a dále RSS profily, do kterého směřují data z RSS a dále data z Multi-sensorové lokalizace. Pro RSS lokalizaci se využívá metody založené na fingerprinting. Nejprve se vytvoří databáze referenčních bodů se známou polohou a RSS z bezdrátových AP (access points – přístupové body). Síla signálu (RSS) v místě referenčního bodu přijímaného z AP. Dalším krokem je určení polohy, které se provádí skrze hledání nejvyšší shody mezi měřeným RSS a RSS v databázi. K tomu je nutné znát eukleidovskou vzdálenost mezi měřeným a referenčním RSS vektorem.

[1] Z angl. received signal strength; tzn. síla přijímaného signálu

Pro určení detekce patra z RSS je nutné znát geometrii signálu z AP z každého patra. Geometrie signálu z AP v patře, kde se robot nachází, by měla být silnější než z AP v ostatních patrech.



Obr. 2.2 – Výsledek datové fúze, 3D lokalizace



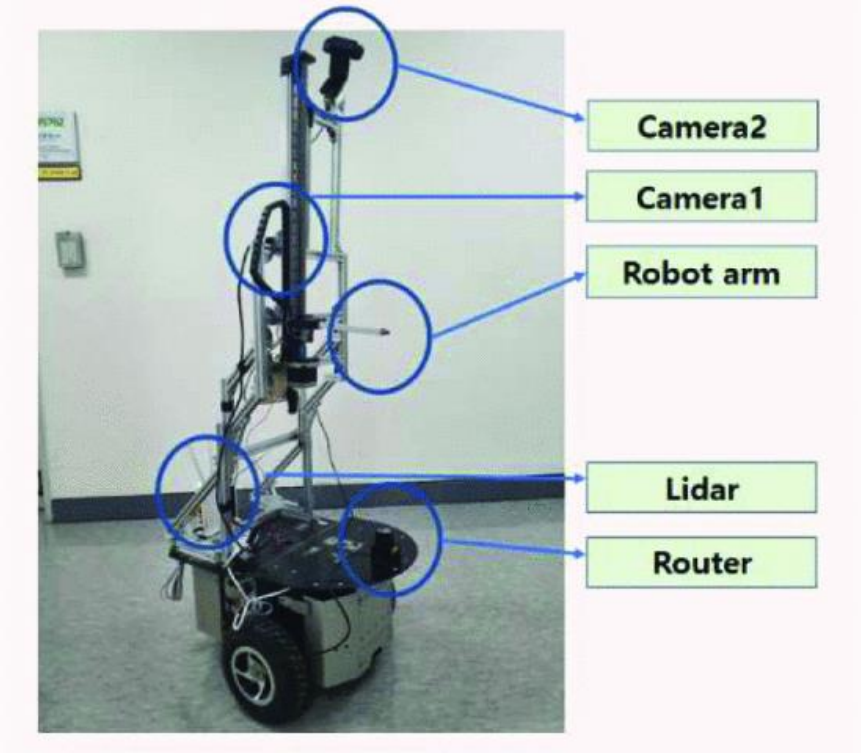
Obr. 2.3 - RSS z AP pro každé podlaží, 3D lokalizace

Obr. 2.2 a Obr. 2.3 byly převzaty ze zdroje [5]. Můžeme zde vidět výsledky z měření při průjezdu jednotlivými patry. Jedná se o měření barometru a magnetometru a dále o měření intenzity signálu. V každém patře bylo nainstalováno 8 AP. Vidíme, že nejvyšší intenzity signálu odpovídají aktuálním patrům. Prezentovaná úspěšnost je 97 %.

2.1.5 Systém vícepodlažní navigace, SungKyunkwan University

[6] Tento systém se zabývá visuální detekcí pater a rozpoznáváním prvků výtahu. Informace o aktuálním patře je získaná z palubního displeje pomocí techniky hlubokého učení.

Na Obr. 2.4 (převzato ze zdroje [6]) můžeme vidět robota, na kterém je systém testován. Robot má 2 kamery, rameno, lidar a router. Kamera 1 slouží k detekci tlačítek, kamera 2 slouží ke sledovaní displeje s čísly pater.



Obr. 2.4 – Testovací robot, Systém vícepodlažní navigace

K určení aktuálního podlaží je použit systém YOLO 9000, který funguje na bázi hlubokých konvolučních neuronových sítí (DCNN). Jedná se o detektor objektů, který dokáže detekovat až 9000 různých kategorií. Úlohou robota bylo použít výtah 1, dojet ze 7. patra do 4. patra, přestoupit na druhý výtah a opět dojet do 7. patra. Dle prezentovaných výsledků byla úspěšnost 80%.

2.2 Metody fúze senzorických dat

[7] Základním cílem při aplikaci fúze dat je redukovat nepřesnost výsledků a zvýšit tak efektivitu klasifikace, detekce a lokalizace objektů. [8] Fúze dat kombinuje data z více senzorů tak, aby dosáhla lepších výsledků, kterých by nebylo možné dosáhnout jedním senzorem.

Metody fúze dat můžeme rozdělit do 3 skupin:

- fúze konkurenční; využití různých typů senzorů k měření stejné fyzikální hodnoty,
- fúze doplňující; každý senzor je využit k měření různých vlastností sledovaného objektu (prostředí),
- fúze spolupracující; správné výsledky senzoru závisí na výsledcích z jiných senzorů. Bez kooperace by byla funkce senzoru nemožná nebo nežádoucí.

Pro potřeby práce bude využívána doplňující fúze, kdy každý senzor snímá jinou vlastnost objektu. V případě detekce aktuálního podlaží při jízdě výtahem pro autonomní robotický systém bude sledováno zrychlení pomocí akcelerometru a informační displej uvnitř výtahové kabiny snímaný kamerou. Jedná se tedy o využití různých typů senzorů na sledování odlišných vlastností sledovaného objektu, z kterých po zpracování získáme informaci o stavu systému. Stavem je myšleno aktuálním podlaží, ve kterém se autonomní robot aktuálně nachází. Systém je tedy plně závislý na měření ze senzorů, které do něho mohou vnášet nepřesnosti. Může se jednat o chybu měření, šum, nízkou citlivost atd. Algoritmy datové fúze pracují s těmito nepřesnostmi s využitím teorie pravděpodobnosti. Výsledek je reprezentován rozdělením pravděpodobnosti na základě možných hypotéz. S ohledem na následné využití teorie pravděpodobnosti při popisu jednotlivých algoritmů, budou představeny základní vztahy. Značení vychází z knihy Probabilistic robotics [9].

2.2.1 Bayesův filtr

Bayesův filtr je rekursivní algoritmus, který je využíván k odhadu *rozdělení důvěry* systému na základě předchozího výstupu, aktuálního akčního zásahu a měření ze senzorů. Jedná se o obecný algoritmus, na jehož základě jsou odvozeny různé algoritmy odhadující stav systému. V oblasti robotiky je využíván např. pro určení pozice či orientace mobilního robota při využití dat z několika senzorů [10].

S ohledem na následné využití teorie pravděpodobnosti při popisu algoritmu, budou představeny některé důležité vztahy.

Vztahy teorie pravděpodobnosti

Předpokládejme měření senzorů, stavy robotického systému a další proměnné, jako náhodné veličiny. Rovnice 2.1 udává pravděpodobnost, že náhodná veličina X má hodnotu x.

$$p(X = x) \tag{2.1}$$

Sdruženou pravděpodobnost v případě nezávislých proměnných označuje rovnice 2.2. Jedná se o pravděpodobnost, kdy oba jevy nastanou současně, tedy o průnik.

$$p(x, y) = p(x)p(y) \tag{2.2}$$

Podmíněnou pravděpodobnost popisují vztahy v rovnici 2.3. Jedná se o pravděpodobnost jevu x za předpokladu jevu y. Druhý vztah popisuje pravděpodobnost jevu y za předpokladu jevu x, tedy pravděpodobnost inverzní.

$$p(x|y) = \frac{p(x,y)}{\mathrm{p(y)}};\ p(y|x) = \frac{p(x,y)}{\mathrm{p(x)}} \qquad 2.3$$

Na základě těchto vztahů lze odvodit rovnici, která je známá jako *Bayesova věta*.

$$p(x|y) = \frac{p(y|x)\mathrm{p(x)}}{\mathrm{p(y)}} = \eta\, p(y|x)\mathrm{p(x)} \qquad 2.4$$

Kde:

$p(x|y)$ je posteriorní pravděpodobnost; podmíněná pravděpodobnost x za předpokladu y,

$p(x)$ je apriorní pravděpodobnost x; pravděpodobnost vstupní,

$p(y|x)$ je podmíněná pravděpodobnost y za předpokladu x; označovaná také jako inverzní,

$p(y)$ je apriorní pravděpodobnost y; jedná se o pravděpodobnost, která je nezávislá. To znamená, že hodnota pravděpodobnosti bude pro jakoukoli hodnotu x stejná. Proto je pravděpodobnost $p(y)^{-1}$ značena jako normalizační proměnná η.

Tento vztah je zásadní pro práci s podmíněnými pravděpodobnostmi. Jedná se o využití inverzní pravděpodobnosti. Definujme x jako hledaný stav a y jako měření. Posteriorní pravděpodobnost udává pravděpodobnost hledaného stavu při aktuálním měření. Tedy např. pravděpodobnost aktuální polohy robotu při konkrétních datech ze senzoru. Aby bylo možné tuto pravděpodobnost určit, je nutné znát přesnost senzoru. Tato přesnost je určena inverzní pravděpodobností, kdy je určena pravděpodobnost měření za předpokladu, že x byl hledaný stav. Tedy je hledána pravděpodobnost, s jakou dokáže senzor určit definovaný stav.

V případě, že proměnné x a y nejsou na sobě závislé, je podmíněná pravděpodobnost

$$p(x|y) = \frac{p(x)p(y)}{\mathrm{p(y)}} = p(x) \qquad 2.5$$

Pro tyto nezávislé jevy lze odvodit vztah nazývaný jako Věta o úplné pravděpodobnosti. Pro spojité případy platí rovnice 2.6. Pro diskrétní případy je integrál nahrazen sumou přes všechna y.

$$p(x) = \int p(x|y)p(y)dy \qquad 2.6$$

$$p(x) = \sum_i p(x|y_i)p(y_i) \qquad 2.7$$

Rozdělení důvěry

Důležitým konceptem, který je zaveden je pojem důvěra[2], označení *bel*. Jedná se o aktuální odhad stavu systému, který je reprezentován hustotou pravděpodobnosti přes všechny možné stavy. Jedná se o interní znalost, důvěru robota ve stav prostředí.

Rozdělení důvěry je vypočteno přes podmíněné rozdělení pravděpodobnosti. Příkladem může být lokalizace mobilního robota, kdy je na základě předchozí polohy a měření vypočítána hustota pravděpodobnosti přes všechny možné polohy, ve kterých se robot může nacházet.

Výpočet důvěry $\mathrm{bel}(x_t)$ je popsán rovnicí 2.8.

$$\mathrm{bel}(x_t) = p(x_t|z_{1:t}, u_{1:t}) \tag{2.8}$$

kde x_t popisuje stav systému, $z_{1:t}$ jsou všechna předchozí měření a $u_{1:t}$ jsou všechny předchozí akce.

Matematické odvození Bayesova filtru

Odvození Bayesova filtru vychází z výpočtu posteriorní pravděpodobnosti $p(x_t|z_{1:t}, u_{1:t})$, která je získaná na základě posteriorní pravděpodobnosti z kroku předchozího $p(x_{t-1}|z_{1:t-1}, u_{1:t-1})$, na základě akčního zásahu u_t a měření z_t .

V prvním kroku je na posteriorní pravděpodobnost aplikována Bayesova věta.

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) = \eta p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t})\, p(x_t|z_{1:t-1}, u_{1:t}) \tag{2.9}$$

Tento vztah lze zjednodušit, protože aktuální měření z_t není závislé na měřeních $z_{1:t-1}$ ani na akcích $u_{1:t}$. To vyjadřuje rovnice 2.10.

$$p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t}) = p(z_t|x_t) \tag{2.10}$$

Pravděpodobnost $p(x_t|z_{1:t-1}, u_{1:t})$ je označována jako $\overline{bel}(x_t)$, jedná se o odhad rozdělení důvěry. Tento krok je v algoritmu označovaný jako predikce.

Výsledná důvěra systému pro stav x_t je

$$bel(x_t) = \eta p(z_t|x_t)\overline{bel}(x_t) \tag{2.11}$$

K získání kompletního algoritmu je nutné vyjádřit odhad rozdělení důvěry $\overline{bel}(x_t)$. Je využita věta o úplné pravděpodobnosti

$$\begin{aligned} \overline{bel}(x_t) &= p(x_t|z_{1:t-1}, u_{1:t}) \\ &= \int p(x_t|z_{1:t-1}, u_{1:t}, x_{t-1})\, p(x_{t-1}|z_{1:t-1}, u_{1:t-1}) dx_{t-1} \end{aligned} \tag{2.12}$$

V případě, že je splněn Markovův předpoklad, stav x_t závisí pouze na předchozím stavu x_{t-1} a akčním zásahu u_{t-1}. Pravděpodobnost $p(x_{t-1}|z_{1:t-1}, u_{1:t-1})$ odpovídá důvěře $bel(x_{t-1})$ z předchozího kroku.

[2] z angl. belief

Po této úpravě je výsledek

$$\overline{bel}(x_t) = \int p(x_t|u_{t-1}, x_{t-1})\, bel(x_{t-1}) dx_{t-1} \qquad 2.13$$

Spojitý Bayesův filtr

Na Obr. 2.5 (převzato z [9]) můžeme vidět pseudokód algoritmu Bayesova filtru, který byl sestaven na základě matematického odvození. V případě lokalizace robota je vstupem akční zásah u_t a měření senzoru poskytující informaci o ujeté vzdálenosti z_t. Cílem je určit rozdělení důvěry bel v aktuální polohu x. Vstupem do algoritmu je také výsledek z předchozího kroku $bel(x_{t-1})$.

Rozdělení důvěry bel je počítáno přes všechny možné stavy x v aktuálním čase t. Toto reprezentuje smyčka v řádku 2, kdy v každé iteraci je vypočtena hodnota důvěry odpovídající konkrétnímu stavu x_t. Jako první je vypočten odhad důvěry $\overline{bel}$ pro stav x_t. Jedná se v případě spojitého rozdělení o výpočet integrálu přes všechny stavy x z předchozího kroku $t-1$. Pravděpodobnost $p(x_t|u_t, x_{t-1})$ vyjadřuje pravděpodobnost aktuálního stavu za předpokladu aktuálního akčního zásahu u_t a předchozího stavu x_{t-1}, která je vynásobena příslušnou hodnotou důvěry v předchozí stav x_{t-1}. Tento krok je nazýván *predikce.*

1: **Algorithm Bayes_filter(**$bel(x_{t-1}), u_t, z_t$**):**
2: for all x_t do
3: $\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx$
4: $bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$
5: endfor
6: return $bel(x_t)$

Obr. 2.5 – Základ algoritmu Bayesova filtru

V druhém kroku, který je označován jako *korekce*, je prvotní odhad důvěry $\overline{bel}(x_t)$ vynásoben inverzní podmíněnou pravděpodobností $p(z_t|\, x_t)$. Ta určuje, jaká je pravděpodobnost, že měření z_t probíhalo ve stavu x_t. Symbol η reprezentuje normalizaci. Výsledkem je hodnota důvěry bel pro konkrétní stav x_t. Běh algoritmu je ukončen v případě, kdy jsou vypočteny hodnoty důvěry pro všechny stavy. Výsledkem je rozdělení důvěry $bel(x_t)$.

Diskrétní Bayesův filtr

V případě diskrétních hodnot musí být Bayesův filtr upraven. Na Obr. 2.6 (převzato z [9]) můžeme vidět pseudokód diskrétního Bayesova filtru.

Vstupem do algoritmu je akční zásah u_t, měření z_t a také všechny hodnoty pravděpodobnosti $\{p_{k,t-1}\}$ z předchozího kroku v čase $t-1$. Cílem je aktualizovat rozdělení pravděpodobnosti (důvěry) $\{p_{k,t}\}$. Příkladem může být lokalizace robota, při jízdě výtahem ve vícepodlažní

budově. Kdy akční zásah u_t je reprezentovaný změnou patra při jízdě výtahem snímaným akcelerometrem, z_t je měření sledující informační displej uvnitř výtahu a x_k je podlaží.

1: **Algorithm Discrete_Bayes_filter($\{p_{k,t-1}\}, u_t, z_t$):**
2: for all k do
3: $\bar{p}_{k,t} = \sum_i p(X_t = x_k \mid u_t, X_{t-1} = x_i)\, p_{i,t-1}$
4: $p_{k,t} = \eta\, p(z_t \mid X_t = x_k)\, \bar{p}_{k,t}$
5: endfor
6: return $\{p_{k,t}\}$

Obr. 2.6 – Algoritmus diskrétního Bayesova filtru

Výsledné rozdělení pravděpodobnosti $\{p_{k,t}\}$ je počítáno přes všechny stavy x_k v aktuálním čase t. Toto reprezentuje smyčka for v řádku 2, kdy v každé iteraci je vypočtena hodnota důvěry odpovídající konkrétnímu podlaží s indexem k.

Jako první je vypočten odhad pravděpodobnosti (důvěry) $\overline{p_{k,t}}$. V případě diskrétního rozdělení se jedná o výpočet sumy přes všechny stavy x_i z předchozího kroku $t-1$. Pravděpodobnost $p(x_k|u_t, x_i)$ vyjadřuje pravděpodobnost, že robot se nachází v podlaží x_k za předpokladu akčního zásahu u_t a předchozího stavu x_i. Za akční zásah lze považovat jízdu výtahem, tedy velikost změny podlaží a předchozí stav x_i reprezentuje podlaží, ve kterém se robot nacházel v minulém časovém okamžiku. Tato podmíněná pravděpodobnost je vynásobena příslušnou hodnotou pravděpodobnosti $p_{i,t-1}$ pro předchozí stav i. Tento krok je nazýván *predikce.*

V druhém kroku, který je označován jako *korekce*, je prvotní odhad důvěry $\overline{p_{k,t}}$ vynásoben inverzní podmíněnou pravděpodobností $p(z_t|\, x_k)$. Ta určuje, jaká je pravděpodobnost, že měření z_t probíhalo ve stavu x_k. Symbol η reprezentuje normalizaci. Výsledkem je hodnota důvěry $p_{k,t}$. Algoritmus je ukončen v případě, kdy jsou vypočteny hodnoty důvěry pro všechny stavy. Výsledkem je rozdělení pravděpodobnosti $\{p_{k,t}\}$ pro všechny možné stavy.

2.2.2 Kalmanův filtr

Kalmanův filtr, dále KF, je matematický algoritmus, který umožňuje odhadovat stavy lineárních systémů na základě stavového modelu a měření ze senzorů. [11] Výsledek KF může být aktuální i budoucí odhad stavu systému, který je statisticky optimální. Jedná se o implementaci Bayesova filtru ve spojitém prostoru. Podmínkou pro použití KF je, že všechny veličiny mají Gaussovo rozdělení pravděpodobnosti, z toho důvodu se tato skupina nazývá Gaussovské filtry.

Algoritmus KF má dvě části, predikci a korekci. V predikci je odhadován stav systému na základě matematického modelu. Aproximovaný výsledek se poté porovná s výsledkem z reálného měření a provede se korekce. Z matematického pohledu se jedná o kombinaci střední

hodnoty a rozptylu měření se střední hodnotou a rozptylem modelu. Výsledkem je optimální odhad stavu systému s co nejmenším rozptylem.

Popis KF vychází ze stavového modelu, který popisuje rovnice 2.14 a z rovnice měření 2.15.

$$x_{k+1} = \mathrm{A}\mathrm{x}_\mathrm{k} + \mathrm{B}u_k + w_k \tag{2.14}$$

Kde x označuje vektor stavů, A je matice soustavy, B je matice vstupů a w_k reprezentuje šum modelu.

$$y_{k+1} = Hx_k + v_k \tag{2.15}$$

Kde y_{k+1} označuje vektor měření, H je matice měření a v_k reprezentuje šum senzoru. Matematické odvození rovnic KF lze nalézt v [9]. Výsledné matematické rovnice predikce a korekce jsou popsané v následující části.

Predikce

$$\hat{x}_{k+1}^{-} = \mathrm{A}\hat{x}_k^{+} + \mathrm{B}u_k \tag{2.16}$$

Kde $\hat{x}_{k+1}^{-}$ je predikce stavového vektoru, $\hat{x}_k^{+}$ je aktuální stavový vektor, u_k je vektor vstupů.

$$P_{k+1}^{-} = \mathrm{A}P_k^{+}\mathrm{A}^T + \mathrm{V} \tag{2.17}$$

P_{k+1}^{-} je kovarianční matice odhadnutých stavů a určuje rozptyl odhadnutých stavů. Matice P_k^{+} je kovarianční matice aktuálních stavů. Matice V reprezentuje procesní šum modelu. Jedná se o parametr, který určuje důvěru ve správnost modelu.

Korekce

$$\hat{x}_{k+1}^{+} = \hat{x}_{k+1}^{-} + K_{k+1}\tilde{y}_{k+1} \tag{2.18}$$

$$P_{k+1}^{+} = P_{k+1}^{-} - K_{k+1}\mathrm{H}P_{k+1}^{-} \tag{2.19}$$

Kde

$$\tilde{y}_{k+1} = y_{k+1} - H\hat{x}_{k+1}^{-} \tag{2.20}$$

$$\mathrm{S} = HP_{k+1}^{-}\mathrm{H}^T + W \tag{2.21}$$

$$K_{k+1} = P_{k+1}^{-}H^T S^{-1} \tag{2.22}$$

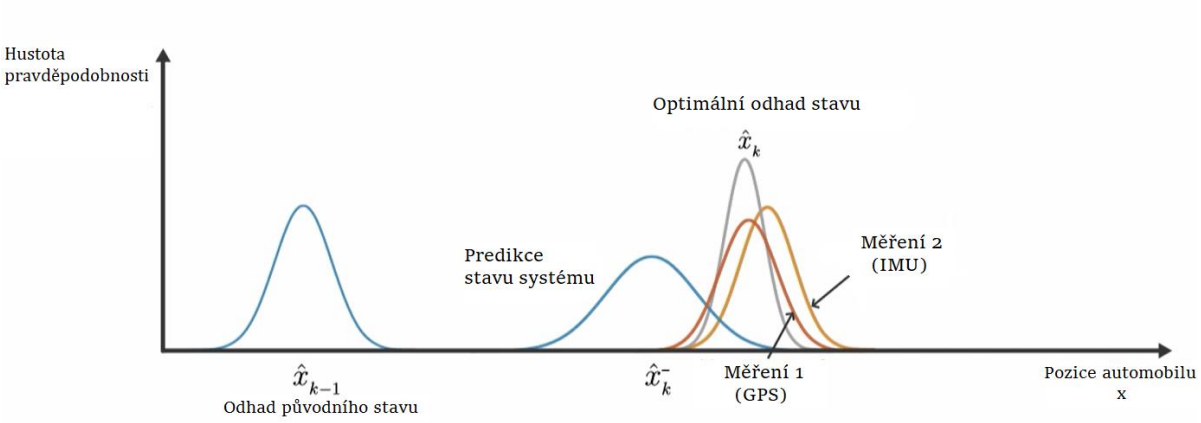
Kde $\hat{x}_{k+1}^{+}$ je optimální odhad (korekce), $\hat{x}_{k+1}^{-}$ je počáteční odhad (predikce z modelu), K_k je Kalmanovo zesílení a y_{k+1} je vektor měření senzoru. H je matice měření, která určuje, jaký stav je měřen. Matice W reprezentuje šum senzoru neboli rozptyl měření.

Tento přístup lze lehce upravit na fúzi dat ze senzorů. Jedná se o rozšíření vektoru měření, matice měření a Kalmanova zesílení pro více hodnot. Příkladem datové fúze je odhad pozice

automobilu na základě modelu a měření. Jedná se o měření senzoru 1 (GPS) a měření senzoru 2 (Inerciální měřící jednotky – IMU). Fúze dvou senzorů by vypadala takto:

$$\hat{x}_{k+1}^{+} = \hat{x}_{k+1}{}^{-} + K_k(y_{k+1} - H_{k+1}\hat{x}_{k+1}{}^{-}) \qquad 2.23$$

Kde $\hat{x}_{k+1}^{+}$ je optimální odhad, je $\hat{x}_{k+1}^{-}$ počáteční odhad (predikce z modelu), K_k je Kalmanovo zesílení, y_k je vektor měření senzorů 1 a 2, H je matice měření, která určuje, jaký stav je měřen. Tato situace je zobrazena na Obr. 2.7 (převzato a upraveno dle [12]), kde můžeme vidět závislost hustoty pravděpodobnosti na sledované pozici. Výsledkem je optimální odhad s nejmenším rozptylem.



Obr. 2.7 – Vizualizace fúze, optimální odhad stavu

Výhodou KF je především to, že dokáže odhadovat aktuální i budoucí stavy. Algoritmus dokáže spojovat více zdrojů informací. Při jeho použití není nutné měřit všechny požadované veličiny. Také dokáže pracovat s nepřesnostmi v modelu i s nepřesnostmi senzorů a jejich vliv zmenšit. Proto je využíván např. při fúzi dat senzorů pro určení polohy autonomních systémů [5], [13] atd.

Omezení v použití KF je v rozdělení pravděpodobnosti, kdy pro korektní použití musí být splněn předpoklad, že všechny veličiny mají normální rozdělení a nulovou střední hodnotu, tj. Gaussovo rozdělení pravděpodobnosti. Základní typ KF navíc funguje jen pro lineární systémy, je však možné rozšířit jeho použití i na nelineární systémy, viz. EKF.

2.2.3 Částicový filtr

[9] Jedná se o neparametrickou implementaci Bayesova filtru. Nejsou založené na pevné funkcionální formě posteriorní pravděpodobnosti jako Gaussovské filtry, ale aproximují posteriorní pravděpodobnosti konečným počtem hodnot, které jsou pevně spojené s určitou oblastí stavového prostoru. [14] *Částicový filtr* na rozdíl od KF dokáže najít řešení i pro problémy, které nejsou lineární nebo nemají Gaussovo rozdělení pravděpodobnosti. Nevýhodou vůči KF mohou být vyšší výpočetní nároky.

Částicový filtr se skládá ze třech hlavních částí:

- vzorkování,

- váhování,
- převzorkování.

Základní myšlenkou je reprezentace posteriorní pravděpodobnosti $bel(x_t)$ pomocí sady částic X_t. Částice je definována jako

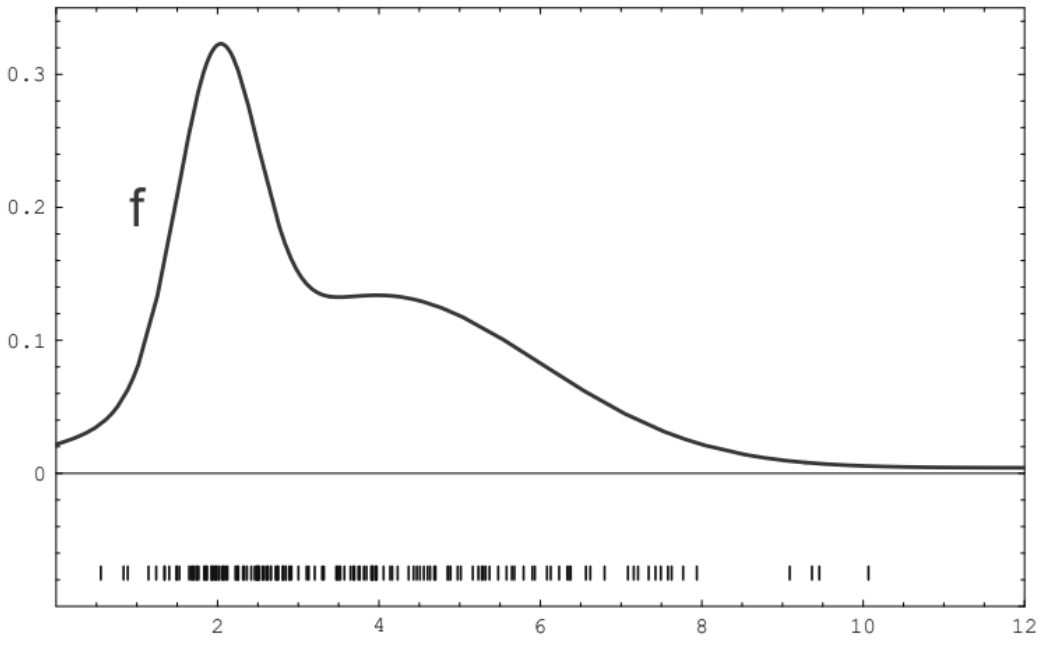
$$X_t := x_t^{[1]}, x_t^{[2]}, \dots, x_t^{[M]} \qquad 2.24$$

kde M odpovídá počtu částic v sadě.

Částice tedy přibližně odpovídá hodnotě důvěry $bel(x_t)$.

$$x_t^{[m]} \sim p(x_t | z_t, u_{t-1}) \qquad 2.25$$

Tam, kde je hodnota pravděpodobnosti vysoká, je také vysoká hustota částic. Kde je naopak hodnota pravděpodobnosti nízká, je také nízká hustota částic v tomto místě. Viz. Obr. 2.8 (převzato ze [9]).



Obr. 2.8 – Vizualizace aproximace funkce *f* pomocí částic

Na základě této představy je zřejmé, že rozložení pravděpodobnosti může mít téměř libovolný tvar.

Základ tohoto algoritmu je v Bayesově filtru, proto je jeho funkcionalita založena opět na rekurzivním přístupu. Algoritmus Částicového filtru je znázorněn na Obr. 2.9 (převzato ze [9]). Vstupem do funkce je set částic z předchozí iterace X_{t-1} spolu s aktuálním vstupem u_t a měřením z_t . Na základě vstupních hodnot jsou vygenerované nové částice, které odpovídají rozdělení pravděpodobnosti x_t za předpokladu řídícího vstupu u_t a předchozího stavu x_{t-1}. Tento krok lze označit za *predikci*.

Další fází je váhování částic, kdy na základě informací z měření jsou vypočítané váhy $w_t^{[m]}$. Na základě výpočtu vah jsou více pravděpodobné částice posíleny, naopak méně pravděpodobné jsou zeslabeny. Jedná se tedy o *korekci*.

Poslední, velmi důležitou fází je nové rozmístění částic, *převzorkování*. Částice jsou rozmístěny dle výsledné posteriorní pravděpodobnosti $bel(x_t)$ z předešlého kroku. Nové částice jsou vždy rozmístěny do míst s vysokou hodnotou posteriorní pravděpodobnosti.

1: **Algorithm Particle_filter**($\mathcal{X}_{t-1}, u_t, z_t$):
2: $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
3: for $m = 1$ to M do
4: sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$
5: $w_t^{[m]} = p(z_t \mid x_t^{[m]})$
6: $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7: endfor
8: for $m = 1$ to M do
9: draw i with probability $\propto w_t^{[i]}$
10: add $x_t^{[i]}$ to $\mathcal{X}_t$
11: endfor
12: return $\mathcal{X}_t$

Obr. 2.9 – Algoritmus Částicového filtru

Částicový filtr je využíván tam, kde se objevují veličiny, které nemají normální rozdělení pravděpodobnosti.

2.3 Klasifikace vzorů v digitálním obrazu

Klasifikace obrazů a práce s digitálním obrazem je jednou ze tří hlavních částí práce. Proto je nezbytné se seznámit s možnostmi a přístupy, které jsou v této oblasti používány. Problematika klasifikace a rozpoznávání vzorů v digitálním obrazu je velmi široká. Avšak metody dle [15] lze rozdělit na dva směry. Jedná se o

- klasický přístup (CV[3]),
- hluboké učení.

Klasický přístup

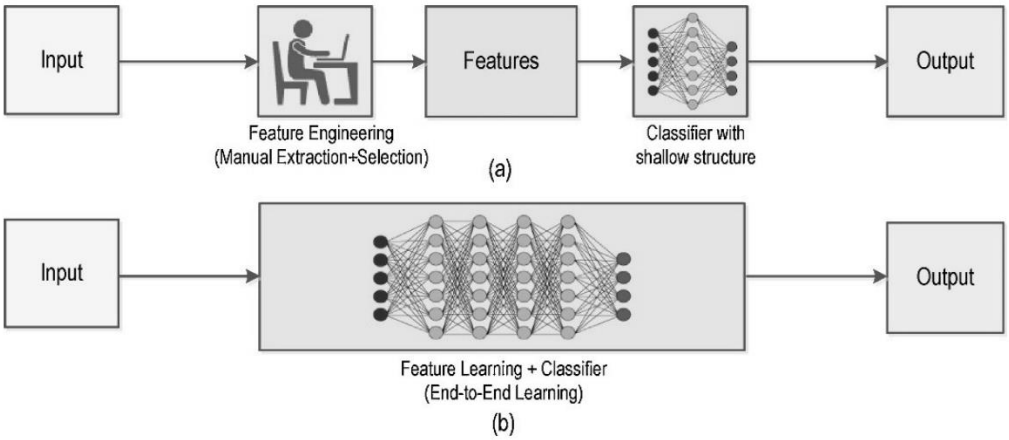
[15] Klasický přístup je založený na ověřených technikách využívajících *extrakci důležitých rysů*[4]. V obrazu jsou nalezeny zajímavé oblasti, které jsou podrobeny klasifikaci. Tento krok může obsahovat techniky jako např. detekce hran nebo prahová segmentace. Ze znaků je

[3] Z angl. Computer vision, počítačové vidění
[4] Z angl. feature *extraction*; *feature* lze v kontextu přeložit jako rys; tedy extrakce „důležitých“ rysů

vytvořena definiční sada známá jako *bag-of-words*[5]. V další fázi jsou tyto znaky hledány v ostatních snímcích. Pokud je daný snímek obsahuje, je přiřazen do klasifikační třídy odpovídající tomuto znaku. Nevýhodou klasického přístupu je nutnost definovat, jaké znaky jsou důležité pro konkrétní obraz. Správná klasifikace je závislá na úsudku CV inženýra a na pokus-omyl přístupu. Výsledek je závislý na mnoha parametrech, které závisí na úsudku konkrétního člověka.

Hluboké učení

Jedná se o podoblast strojového učení. Tato funkce imituje chování lidského mozku při zpracování informací. Význam metody hlubokého učení stoupl především s vývojem výkonných počítačů, které umožňují vyšší rychlost učení. Jedná se o koncept end-to-end[6].



Obr. 2.10 – Klasifikace pomocí a) klasického přístupu, b) konceptu hlubokého učení

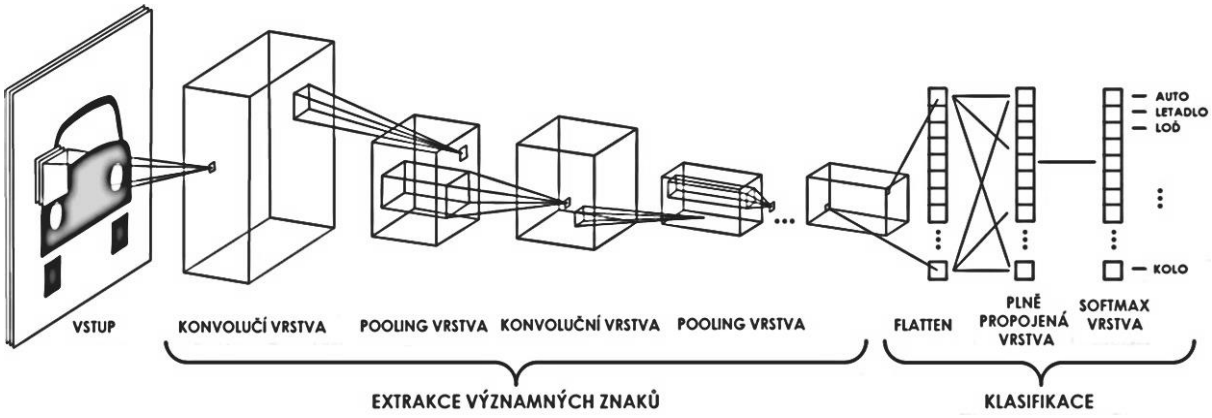
Jedním ze základních algoritmů pro využití hlubokého učení jsou konvoluční neuronové sítě (CNN). Uvažujme CNN za uzavřený systém, do kterého vedou vstupy a z něj vychází výstupy. Vstupem je např. obraz, výstupem je třída klasifikace. To znamená, že uvnitř neuronové sítě se odehraje vše od extrakce rysů po klasifikaci do tříd. Na Obr. 2.10 (převzato z [15]) můžeme vidět rozdíl mezi klasickým přístupem a zmiňovaným přístupem hlubokého učení. Na rozdíl od klasického přístupu se k extrakci rysů využívá jednotlivých vrstev sítě, kdy vyšší vrstvy identifikují rysy jako např. čísla nebo obličeje, nižší vrstvy identifikují různé rysy obsažené v obrazu, např. hrany. V další části budou popsané složitější CNN, možnosti implementace a jejich použití pro úlohu klasifikace.

2.3.1 Konvoluční neuronové sítě (CNN)

Jedná se o podmnožinu umělých neuronových sítí (ANN). Neuronová síť se skládá z perceptronů, které jsou spojeny do vrstev. Matematický popis perceptronu a jednoduchých neuronových sítí nejsou cílem tohoto textu, popis zde [16]. CNN byly vyvinuty pro práci s vícedimenzionálními signály. Jsou založeny na konvolučních vrstvách, které jak již bylo

[5] *bag-of-words* lze chápat jako sada definovaných znaků
[6] *end-to-end;* od začátku do konce, vše je realizováno v CNN

zmíněno, slouží k extrakci významných rysů v obrazu. Příklad CNN můžeme vidět na Obr. 2.11 (upraveno dle [17]).



Obr. 2.11 – Vizuální představa CNN

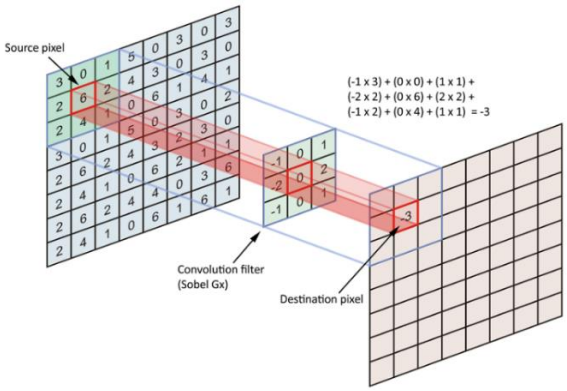
Vidíme, že síť se skládá ze dvou funkčních celků, kterými jsou

- extrakce významných rysů,
- klasifikace.

Nejprve jsou nalezeny významné rysy v obraze, které slouží jako vstup do klasifikační části sítě. Výstupem je rozdělení pravděpodobnosti mezi třídy klasifikace. Jednotlivé vrstvy budou detailněji rozebrané v další části.

Konvoluční vrstva

Je založena na konvoluci. Konvoluce je matematická operace mezi dvěma funkcemi. Kde první funkcí je myšleno konvoluční jádro (kernel) a druhou vstupní funkce. V případě práce s digitálním obrazem je využita 2D konvoluce. Výstupem je extrakce významných rysů v obraze.



Obr. 2.12 – Vizuální představa konvoluce

Funkci konvoluční vrstvy demonstruje Obr. 2.12 [18]. Obraz lze interpretovat jako matici pixelů. K procesu konvoluce se musí vytvořit konvoluční jádro nazývané jako konvoluční filtr, hodnoty uvnitř filtru jsou nazývané váhy. Filtr se posouvá po vstupním obraze a v každém kroku se provede skalární součin s částí, se kterou se překrývá.

Konvoluční vrstva má několik základních parametrů, které musí být v případě jejího využití definovány. Jedná se o:

- počet filtrů; udává hloubku sítě, každému filtru odpovídá jedna výstupní 2D matice,
- velikost konvolučního jádra; výška a šířka filtru v pixelech,
- posunutí (stride); udává velikost posunutí konvolučního jádra mezi dvěma kroky konvoluce v pixelech.

Důležitou vlastností sítě je sdílení vah. Jedná se o vlastnost, která umožňuje používat pro všechna posunutí filtru stejné váhy. Mějme konvoluční vrstvu, která se skládá z 5 filtrů o rozměrech [2x2] vstup má rozměry [3x3x1]. Pak výstup sítě bude mít dimenzi [2x2x5]. Počet neuronů $3 \cdot 3 \cdot 5 = 45$, počet vah pro každý neuron $2 \cdot 2 \cdot 1 = 4$.

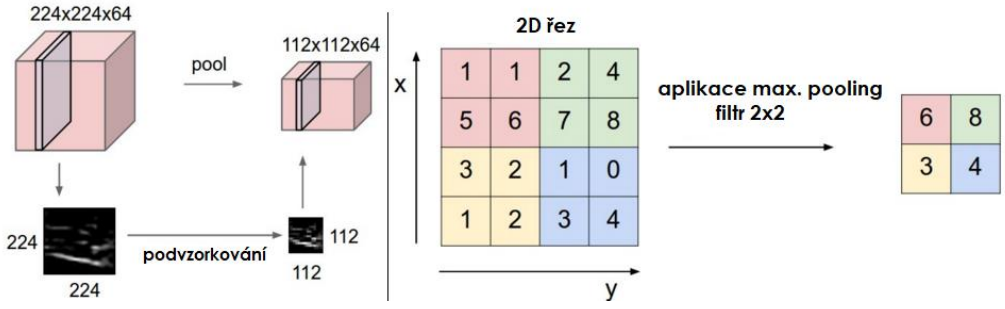
V případě nesdílených vah je jejich počet $45 \cdot 4 = 180$. V případě sdílených vah $4 \cdot 5 = 20$. Tato vlastnost výrazně sníží počet parametrů v konvoluční vrstvě.

Pooling vrstva

Pooling vrstva[7] redukuje prostorovou dimenzi, avšak zachovává hloubku sítě. [16] Vrstva poskytuje tyto výhody:

- snižuje výpočetní náročnost,
- redukuje počet parametrů a snižuje nebezpečí *přetrénování*[8],
- umožňuje částečnou prostorovou invarianci.

Pooling se provádí tak, že redukuje velikost vstupu pomocí filtru (nejčastěji 2x2), který se posouvá podobně jako při konvoluci a výstupem je číslo v závislosti na zvolené metodě poolingu. Nejčastější metodou je max-pooling, tedy výběr maxima ze zvolené oblasti. Vizuální představa je zobrazena na Obr. 2.13 (upraveno dle [16]).



Obr. 2.13 – Grafické zobrazení Max-pooling vrstvy

[7] Nemá český ekvivalent. Často se nazývá také jako *downsampling*, tzn. podvzorkování.
[8] Z angl. *overfitting*; nadměrná specializace na data, na kterých se provádělo učení.

Flatten vrstva

Flatten[9] vrstva převádí vícedimenzionální vstup na jednodimenzionální výstup. Výstupem je tedy sloupcový vektor. Tento krok je nutný pro další využití dat.

Plně propojená vrstva

Základní prvek klasifikační části CNN. V této vrstvě jsou *významné rysy* klasifikovány do tříd.

Architektura vrstvy je taková, že každý neuron aktuální vrstvy N je propojen se všemi neurony předchozí vrstvy M, proto označení plně propojená vrstva. Výstup vrstvy N udává rovnice

$$o_j = f\left(\sum_{i}^{n} w_{ij} o_i + b\right) \qquad 2.26$$

Kde o_j je výstup j-tého neuronu vrstvy N, f je aktivační funkce, o_i je výstup z i-tého neuronu předchozí vrstvy M, w_{ij} označuje váhu mezi těmito neurony a b je bias. Popis a výběr aktivační funkce bude popsán v kapitole zabývající se hyperparametry.

Softmax vrstva

Jedná se o vrstvu, která transformuje výstup plně propojené vrstvy na vektor definovaný počtem klasifikačních tříd s hodnotou v intervalu (0, 1) a celkovým součtem 1. Je tak určeno rozdělení pravděpodobnosti mezi jednotlivé klasifikační třídy.

2.3.2 Trénování sítí

Jedná se o metodu učení s učitelem. Trénink probíhá na základě trénovacích vstupních dat, které jsou označeny v závislosti na příslušnosti k dané klasifikační třídě. Toto označení je nazýváno *štítek*[10]. Je tak vytvořena dvojice vstup a k němu požadovaný výstup. Cílem učení je nalézt parametry sítě, při kterých je minimální chyba predikce na výstupu. Trénování má 2 hlavní části. Jedná se o *dopředné šíření*, kdy je vypočtena chyba predikce při aktuálním nastavení vah a dále *zpětné šíření chyby*, kdy jsou upraveny váhy tak, aby byla výsledná chyba predikce minimalizována. Rozlišujeme trénovací a validační datovou sadu, které jsou použity při trénování sítě. Trénovací sada je přímo použita ve fázi učení. Validační sada je použita na zhodnocení natrénované sítě na základě evaluačních metrik, které popisuje kapitola 2.3.3.

Hyperparametry

Jedná se o proměnné, které ovlivňují vlastnosti sítě a postup učení. Jsou zmíněné ty, které jsou využité při tréninku neuronové sítě v další části práce.

- Výběr aktivační funkce; Aktivační funkce definuje výstup na základě daného vstupu. Příkladem může být funkce ReLU. V případě záporného vstupu jsou výstupní hodnoty nastaveny na 0. V případě kladného vstupu jsou hodnoty zachovány. Závislost je v kladné části lineární. Detailní popis zde [17].

[9] Nemá český ekvivalent. V překladu zploštit, zplacatit.
[10] Z angl. label

- Změna počtu plně propojených vrstev a jejich velikosti,
- Nastavení konstanty učení. Jedná se o parametr úměrnosti, který přímo ovlivňuje velikost změny vah při procesu učení. Jinak řečeno určuje rychlost, s jakou se síť adaptuje na daný problém.
- Počet epoch; epocha je definována jako průchod všech trénovacích dat neuronovou sítí. V případě, že je nastaveno např. 10 epoch, každý prvek v datové sadě je využit ve fázi učení celkem 10krát.

Transfer learning

Technika učení, kdy je využit již natrénovaný model neuronové sítě, který je použit při tréninku na nové datové sadě. Jedná se o využití části modelu, která v obrazu extrahuje *významné znaky*. Proto je fáze učení realizována pouze na klasifikační vrstvě, parametry ve zbylých hlubších vrstvách nejsou v průběhu učení nijak měněny. Tímto přístupem je možné s menším počtem trénovacích dat dosáhnout stejné nebo vyšší úspěšnosti klasifikace, než které je možno dosáhnout při trénování nového modelu.

Fine Tuning

Fine tuning technika je využívána pro zlepšení výsledné přesnosti modelu. Jedná se o přístup, kdy jsou v průběhu učení měněny váhy i v hlubších vrstvách modelu. Díky tomuto kroku jsou měněny parametry v konvolučních vrstvách a tím jsou extrahovány *významné znaky* specifické pro danou úlohu.

2.3.3 Evaluace učení

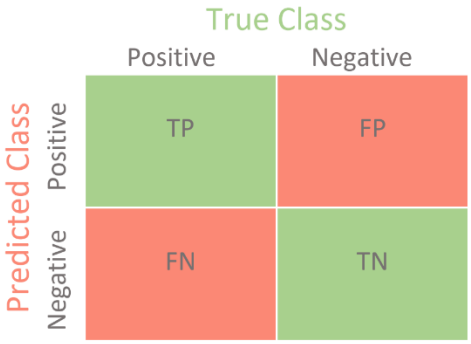
Pro evaluaci učení jsou posuzovány evaluační metriky. Existuje několik metrik, podle kterých je možné hodnotit úspěšnost učení neuronové sítě. Mezi ně patří *matice záměn*[11], accuracy, precision a recall.

Matice záměn

Matice záměn poskytuje evaluace učení v závislosti na klasifikačních třídách. Srovnává se predikce sítě a skutečné označení tříd, které je určené pro daný obraz. Je tak možné určit, které prvky jsou klasifikované správně, a u kterých je klasifikace chybná. V případě chybné klasifikace je určeno, mezi jakými třídami klasifikace došlo k záměně. Příklad binární matice záměn je na Obr. 2.14 (převzato a upraveno dle [19]).

Matice označuje binární klasifikaci, kde je možné určit positivní a negativní klasifikační třídu. Ve sloupcích jsou skutečné označení klasifikačních tříd, v řádcích jsou predikované klasifikační třídy. Správné klasifikace označují hodnoty TP (true positive – pravdivě positivní) a TN (true negative – pravdivě negativní). Nesprávné klasifikace jsou označené jako FP (false positive – falešně positivní) a FN (false negative – falešně negativní).

[11] Z angl. confusion matrix



Obr. 2.14 – Příklad matice záměn pro binární klasifikaci

Na základě tohoto označení je možné vypočítat další hodnotící metriky. Následující metriky je možné všechny přeložit do českého jazyka jako přesnost. Nejedná se však o stejné hodnoty, jejich výpočet se však liší. Proto bude pro větší přehlednost zachován anglický název.

Accuracy

Výpočet této metriky je popsán rovnicí

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.27}$$

Jedná se tak o poměr všech pravdivých klasifikací ku sumě všech klasifikací.

Precision

Další metrikou je precision. Jedná se o poměr pravdivých pozitiv a součtu pravdivých a falešných pozitiv.

$$A = \frac{TP}{TP + FP} \tag{2.28}$$

Recall

Poslední popsanou metrikou je recall. Jedná se o poměr pravdivých pozitiv a součtu pravdivých pozitiv a falešných negativ.

$$A = \frac{TP}{TP + FN} \tag{2.29}$$

V následujícím textu bude jako *přesnost* označována metrika *accuracy*.

2.3.4 Výběr a použití klasifikačních CNN

Při implementaci CNN lze zvolit následující postupy.

- navržení vlastní sítě,

- použití známé architektury,
- použití předtrénované sítě.

Možností využití známých architektur či předtrénovaných sítí je velké množství. Vyvíjet vlastní síť tedy pro většinu aplikací nedává smysl. Pro adekvátní použití známých architektur a předtrénovaných sítí je však nutné zmíněné sítě prostudovat, zjistit možnosti jejich využití, zhodnotit parametry a další hodnotící kritéria.

Prvním a velmi důležitým parametrem je použití. Různé architektury sítí jsou optimální pro různé použití a úkoly. Tato práce se zabývá klasifikací obrazových dat, z toho důvodu budou vybírány architektury a předtrénované sítě, které jsou vhodné pro klasifikaci obrazových dat.

Tuto skupinu lze rozdělit na dvě části, které se mohou vzájemně prolínat. Jedná se o sítě, které jsou vyvíjené tak, aby je bylo možné použít pro techniku transfer learning, popsanou v další části.

Mezi takovéto sítě patří např.:

- ResNet
- MobileNet
- VGG
- Inception
- ShuffleNet_V1
- EfficientNet

Dalším možným přístupem je využít síť, která byla vytvořena pro klasifikaci podobných dat. V případě aktuální úlohy se jedná o klasifikaci číslic.

Při výběru sítě jsou důležité tyto parametry:

- přesnost,
- rychlost,
- počet parametrů (velikost sítě).

Pro úlohu klasifikace při využití na robotické platformě je důležité volit síť s vhodným poměrem přesnosti a rychlosti při nízké velikosti sítě. Důležitým měřítkem je také vstupní velikost klasifikovaného obrazu. Poslední věcí je nalezení vhodného nástroje na implementaci.

Sítě pro klasifikaci podobných dat

Jedná se například o modely, které byly natrénovány na datové sadě MNIST[12] nebo na SVHN[13]. Tyto datové sady se skládají z obrázků, kde se objevují číslice 0–9. Jedná se tedy o podobná data, jaká jsou součástí vytvořené datové sady displejů. Z toho důvodu se lze domnívat, že by

[12] MNIST (Modified National Institute of Standards and Technology); jedná se o datovou sadu obsahující černobílé obrázky ručně psaných číslic 0–9 o velikosti 32x32 pixelů

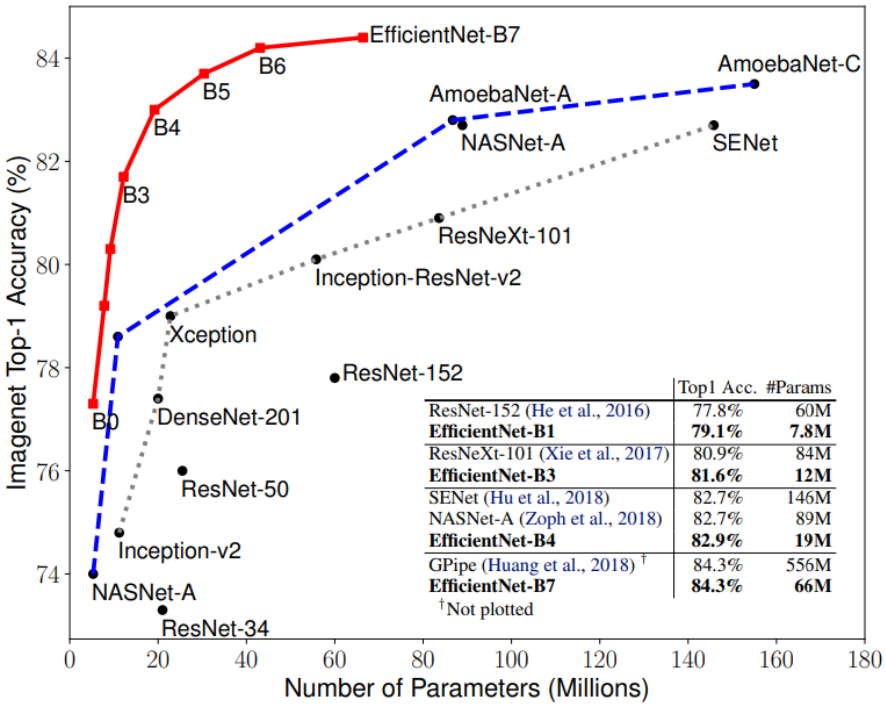
[13] SVHN (The Street View House Numbers); datová sada obsahující barevné obrázky domovních čísel (pro detekci) a jednotlivých vyříznutých číslic (pro klasifikaci) 0-9 o velikosti 32x32 pixelů

sítě natrénované na těchto datových souborech mohly být použitelné i v případě vytvořené obrazové datové sady displejů.

Srovnání CNN vyvíjených na datové sadě ImageNet

Univerzální sítě použitelné na přetrénovaní byly z velké části vyvinuty pro řešení klasifikace obrazových dat z datové sady *ImageNet*. *ImageNet* je obrazová databáze, ve které je aktuálně[14] uloženo přes 14 mil. obrázků. Ty jsou rozděleny do 1000 klasifikačních tříd. Tato databáze je využívaná k vývoji a testování nových modelů pro obrazovou klasifikaci a detekci.

Na Obr. 2.15 můžeme vidět porovnání několika klasifikačních modelů. Jedná se o graf závislosti přesnosti modelu pro datovou sadu ImageNet na počtu parametrů modelu. Z výsledků můžeme vidět, že modely EfficientNet dosahují lepších výsledků v porovnání s modely se stejným počtem parametrů. Jinak řečeno, modely s podobnou přesností mají několikanásobně vyšší počet parametrů. EfficientNet-B7 dosáhla *state-of-the-art*[15] 84,3 % top-1 přesnosti na datové sadě ImageNet.



Obr. 2.15 – Velikost modelu vs. Přesnost na datové sadě ImageNet, srovnání CNN [20]

S ohledem na informace zmíněné výše je předpokládáno využití modelů EfficientNet.

[14] Rok 2021, březen
[15] Nejmodernější, nejlepší v daném oboru

3 FORMULACE PROBLÉMŮ A CÍLE ŘEŠENÍ

Na základě provedené rešerše budou jednotlivé cíle práce blíže specifikovány, budou vybrány vhodné postupy, nástroje a řešení k dosažení těchto cílů.

3.1 Cíle

Hlavním cílem práce je detekce aktuálního podlaží při průjezdu výtahem pomocí fúze dostupných senzorických dat. Předpokládá se využití dat z kamery snímající displej výtahu a dat získaných z akcelerometrického datalogeru. Tato úloha je stěžejní při pohybu robota ve vícepodlažní budově za pomoci výtahu.

Mezi cíle práce patří:

1) Seznámit se se způsoby fúze dat různého typu
2) Vytvořit dostatečnou množinu obrazů informačních prvků výtahu
3) Naměřit průběhy zrychlení při jízdě výtahem pomocí akcelerometru
4) Implementovat vybranou metodu fúze dat
5) Metodu ověřit a vyhodnotit

S ohledem na rešeršní část je nutné tyto cíle blíže specifikovat a nastínit postup realizace.

3.2 Kroky k dosažení cílů

Stěžejním úkolem je zisk dostatečné množiny dat. Jedná se o data, která nejsou veřejně dostupná. Proto se předpokládá série měření v několika různých výtazích.

Další částí je fúze dat. Pro úspěšnou fúzi je nutné použít vhodná řešení pro práci s těmito daty. Základní informace, kterými jsou obrazová data a akcelerometrická data jsou velmi rozdílná a pro využití do fúze je nutné s nimi nejprve pracovat separátně. Úloha se zde musí rozdělit na dva různé algoritmy, jejichž výsledky se v datové fúzi integrují v jeden výsledek.

Proto zde vznikají dílčí cíle práce nezbytné k úspěšnému dokončení.

Jedná se o:

Vývoj a implementaci algoritmu pro zpracování akcelerometrických dat. Předpokládá se výpočet polohy ze známého zrychlení. Algoritmus musí kompenzovat vznik *driftu* při integraci. Důležitým požadavkem je také univerzálnost řešení, kdy je nutné, aby navržené řešení bylo použitelné pro různé výtahy v různých budovách.

Zpracování obrazových dat, výběr a trénink neuronové sítě pro klasifikaci dat. Na tento systém jsou kladeny nároky v přesnosti, robustnosti a rychlosti. Klasifikace musí opět fungovat na různé typy displejů, s různými světelnými podmínkami.

Fúze dat musí umožňovat práci s pravděpodobnostmi, které nemají normální rozložení. Proto se v návaznosti na rešeršní část počítá s využitím Bayesova filtru, který umožňuje práci s různým rozložením pravděpodobnosti. Zároveň je zde možné využít pravděpodobnosti, které jsou získány na základě experimentálního vyhodnocení jednotlivých částí úlohy. Jedná se tedy

o pravděpodobnosti, které jsou založeny na reálných výsledcích z obou zmíněných částí, tedy algoritmus využívající akcelerometrická data a algoritmus využívající obrazová data.

3.3 Zhodnocení rešeršní části

Na základě rešeršní části byla vybrána následující řešení. Jedná se o použití modelu EfficientNet, který splňuje požadavky na přesnost, velikost a rychlost. Tento model bude sloužit jako výchozí bod pro trénování klasifikace obrazových dat z displejů. Na základě rešerše fúze dat byl vybrán Bayesův filtr, který je využíván pro fúzi dat ze senzorů. Výhodou je, že lze kombinovat data, která nemají normální rozdělení pravděpodobnosti. Tato vlastnost je žádaná při práci s pravděpodobnostmi z klasifikátoru.

4 VYTVOŘENÍ DATOVÝCH SAD

Jedním z cílů diplomové práce je získání dostatečné množiny obrazových a akcelerometrických dat při jízdě výtahem. Relevantní a mohutná množina dat je stěžejní pro vytvoření a otestování celého systému. S ohledem na univerzální funkčnost navrženého řešení je nutné, aby tato data byla získána z několika rozdílných výtahů.

4.1 Obrazová data

Pro systém detekce pater je nutné sledovat informaci o aktuálním patře. Tuto informaci poskytuje vnitřní displej kabiny výtahu, který zobrazuje číslo aktuálního podlaží. Z důvodu toho, že nebyla nalezena veřejně dostupná datová sada, která by byla použitelná pro potřeby diplomové práce, bylo nutné takovou sadu vytvořit. Postupu sběru dat se věnuje kapitola 4.1.1, postupu klasifikace obrazových dat do tříd se věnuje kapitola 4.1.2 a příklady displejů jsou ukázané v kapitole 4.1.3. Vytvořená datová sada je pak dále využita při fázi učení konvoluční neuronové sítě pro klasifikaci displejů.

4.1.1 Sběr dat

Problematice sběru obrazových dat se podrobněji věnuje práce M. Černila [21], s kterou tato práce úzce souvisí. Pro lepší orientaci v tématu bude popsán základní koncept sběru dat se zaměřením na displeje s číslem aktuálního podlaží.

Data byla zaznamenávaná na různých mobilních telefonech o minimálním rozlišení 1080x1920 pixelů. Pro potřeby výše zmíněné práce byl vytvořen koncept pro vytvoření videa, které zobrazuje všechny potřebné informační prvky výtahu.

Jedná se o:

- záběr přivolávacího panelu a přední pohled na vnější stranu dveří výtahu,
- záběr na vnitřní tlačítkový ovládací panel s ovládacími prvky výtahu,
- záběr na displej zobrazující aktuální dosažené podlaží v průběhu jízdy nahoru a dolů.

Dalším postupem pro práci s daty je vytvoření anotací. Anotovaná data mohou být následně použita jako vstup pro strojové učení.

4.1.2 Anotace a rozdělení do klasifikačních tříd

Anotaci dat lze z pohledu této práce rozdělit na 2 části. Jedná se o anotování videí se všemi výše popsanými informacemi. Druhou částí je získání částí obrazů, které obsahují displeje a jejich rozdělení podle čísla aktuálního patra, které je na něm zobrazené.

Anotace videí pro detekci

První část opět podrobně popisuje práce M. Černila. K anotaci dat byl použit software CVAT [22]. V něm bylo možné označit všechny sledované informační a ovládací prvky výtahu. Software také umožňuje interpolaci mezi zadanými klíčovými snímky. Tato vlastnost zrychlila

celý proces anotace a byla využívána při všech anotacích. Každému sledovanému prvku byla přiřazena anotační třída s ohraničením, které zobrazuje region odpovídající tomuto prvku.

Rozdělení do klasifikačních tříd

Z anotovaných dat byly poté vygenerovány všechny oblasti, které byly označeny jako displej. Z těchto dat byla vytvořena datová sada, která bude dále využita pro učení klasifikační CNN. Jednotlivá obrazová data displejů byla rozdělena do 15 klasifikačních tříd. Každé klasifikační třídě odpovídá jedna složka, která má v názvu číslo obsažené na displeji. Celkový počet a zastoupení prvků jednotlivých tříd v datové sadě ukazuje Tab. 4.1.

Tab. 4.1 – Informace o zastoupení tříd v datové sadě

Klasifikační třída	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12
Počet prvků	177	897	1404	3003	2339	2292	1519	1751	983	754	579	410	527	360	351

Celkový počet prvků je 17 346. Data byla získána z více než 20 různých výtahových kabin.

4.1.3 Typy displejů

Důsledkem sběru dat z různých výtahů je, že displeje samotné se mohou v jednotlivých výtazích lišit. S přihlédnutím na získaná data lze displeje rozdělit do několika kategorií dle zobrazovaných informací a dle typů.

Dle zobrazovaných informací lze displeje rozdělit na displeje, které jsou popsané a zobrazené v **Tab. 4.2**. Rozděleny jsou do 3 kategorií.

Tab. 4.2 – Typy displejů dle zobrazovaných informací

Číslo aktuálního podlaží	Číslo podlaží a aktuální směr
	

Číslo podlaží, směr a další přídavné informace


Dle typů lze displeje rozdělit do 4 kategorií. Jedná se o displeje sedmi-segmentové, více-segmentové, digitální a dvouciferné. Příklady takových displejů jsou zobrazeny v **Tab. 4.3**.

Tab. 4.3 – Rozdělené displejů dle typu zobrazení

Sedmi-segmentové	Více-segmentové
	
Digitální	S dvouciferným číslem
	

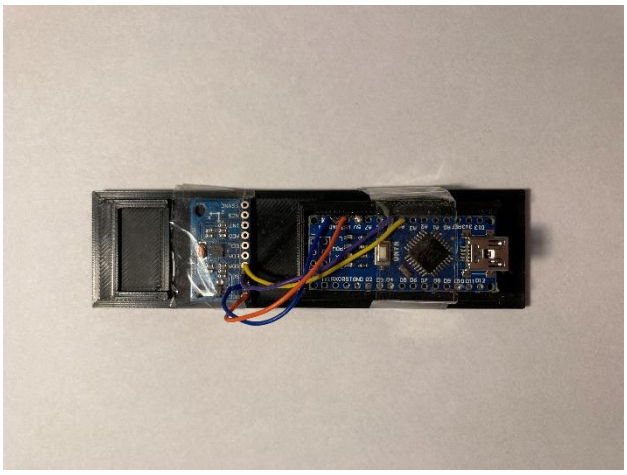
4.2 Množina průběhů zrychlení

Druhou částí tvoření datové množiny je sběr akcelerometrických dat při jízdě výtahem. Jedná se o data, která poskytují informaci o zrychlení v ose z (tj. osa ve které působí gravitační zrychlení). Data v této ose mohou po dalším zpracování poskytnout informaci o aktuální rychlosti a poloze výtahu. Informace o poloze může být využita k určení aktuálního patra.

4.2.1 Měřící zařízení

Data byla získána při jízdě různými výtahy. Měřící zařízení se skládalo z notebooku, desky Arduino NANO a modulu MPU 9250.

Modul v sobě obsahuje akcelerometr, gyroskop, magnetometr a termostat. Jak již bylo zmíněno, z těchto informací je využito zrychlení v ose z. Modul byl připojený k desce Arduino NANO pomocí napájecích a datových kabelů. Zapojení a ukotvení do měřícího stojanu je zobrazeno na Obr. 4.1. V levé části je modul MPU 9250 v pravé části deska Arduino NANO. Detailní popis a parametry senzoru lze nalézt v kapitole 5.1.2.



Obr. 4.1 – Modul MPU 9250 a Arduino NANO

Arduino bylo spojeno s počítačem pomocí kabelu s USB konektorem, který umožňuje full-duplexní sériovou komunikaci. Vzorkovací frekvence měření byla nastavena na 100 Hz. Tento systém byl zvolen proto, že modul s Arduinem je využíván v mobilním robotu. Z toho důvodu je zaručena kompatibilita a jednoduchá implementace na robota.

Na začátku každého měření byl senzor zkalibrován. Jedná se o záznam zrychlení po definovanou dobu. Na základě těchto dat je zjištěna střední hodnota signálu, která odpovídá offsetu senzoru. Tato hodnota je dále odčítána od všech následujících dat. Tím je zajištěno, že se v datech neobjevuje složka tíhového zrychlení.

4.2.2 Sběr dat

Měření probíhala tak, aby data obsahovala jízdu se změnou o jedno až o maximální počet pater, který daný výtah umožňoval. První jízda byla vždy o jedno podlaží nahoru a zpět, poté byly realizované jízdy s vyšší změnou počtu pater. Všechny jízdy byly realizované ve směru nahoru i ve směru opačném.

Data byla ukládána do souboru s koncovkou csv, kde v prvním sloupci byla uložena informace o čase a ve druhém sloupci informace o zrychlení. Všechna měření byla dále doplněna o informaci o průběhu jízdy. Tím se rozumí posloupnost čísel pater, ve kterých výtah zastavil v průběhu měření. Tato informace bude využita k ověření navržených řešení.

4.2.3 Vizualizace měření

V této kapitole je ukázán graf závislosti zrychlení na čase při jízdě výtahem v budově A3 na FSI VUT.

Obr. 4.2 – Graf průběhu zrychlení při jízdě výtahem v budově A2

5 ÚLOHA DETEKCE AKTUÁLNÍHO PODLAŽÍ

V úloze detekce podlaží při jízdě výtahem jsou využívány dva senzory. Jedná se o akcelerometr a o kameru. Nejprve je nutné vyřešit úlohu detekce pomocí jednotlivých senzorů samostatně. Výsledkem bude informace o aktuálním podlaží pro každý systém. S touto informací se bude dále pracovat v datové fúzi.

V této kapitole budou představena jednotlivá řešení. Nejprve bude popsán použitý hardware, tedy kamera a akcelerometr. V kapitole 5.2 bude popsána detekce podlaží s využitím akcelerometrických dat popisujících zrychlení výtahu v průběhu jízdy. V kapitole 5.3 bude popsána detekce podlaží na základě zpracování obrazových dat displejů zobrazujících aktuální podlaží. Dále bude aplikována vybraná metoda fúze dat, která na základě vybrané metody spojí detekované výsledky v jednu informaci s cílem vylepšit přesnost a robustnost celého systému. Popis a použití fúze je v kapitole 5.4.

5.1 Použitý hardware

5.1.1 Kamera

Pro snímání obrazových dat byla použita kamera Microsoft LifeCam HD 3000 s maximálním rozlišením 1280x720 pixelů.

5.1.2 Senzor pro měření zrychlení

Jedná se o modul MPU 9250, který v sobě obsahuje akcelerometr, gyroskop, magnetometr a termostat. Modul je připojen k desce Arduino NANO, která je připojena k mobilnímu robotu nebo k počítači.

Senzor je zobrazen na Obr. 5.1 a jeho připojení k desce je následující. Napájení 3,3V je připojeno na pin VCC, zem je připojena na pin GND, pin s označením SCL (serial clock) zajišťuje časování a pin s označením SDA (serial data) zajišťuje datové spojení mezi senzorem a deskou. Pro komunikaci je využito SPI rozhraní. Detailní dokumentace zde [23]. Součástí senzoru je tříosý 16bitový MEMS akcelerometr s maximálním rozsahem ± 16 g, kde g je gravitační zrychlení. Tento akcelerometr je využíván pro měření.

Obr. 5.1 – Senzor MPU9250

Pro možnosti sériové komunikace byl použit balíček MPU 9250 vytvořený skupinou Bolder Flight Systems, odkaz na soubory a dokumentaci zde [24]. Tento balíček obsahuje funkce pro nastavení modulu a nastavení komunikace mezi deskou Arduino a PC. Upravený kód s vytvořenou knihovnou je v příloze ve složce MPU9250_library. Hodnoty nastavení senzoru popisuje tabulka **Tab. 5.1**.

Tab. 5.1 – Nastavení MEMS akcelerometru

Parametr	Hodnota
Měřící rozsah	± 4 g
Frekvence záznamu dat	100 Hz
Mezní frekvence filtru dolní propust (DLPF)	20 Hz

5.2 Detekce pater na základě akcelerometrických dat

V této kapitole bude popsán algoritmus detekce při využití akcelerometrických dat.

5.2.1 Definice úlohy

Základní myšlenkou detekce pater na základě dat z akcelerometru je využít tato data k určení polohy. Pohyb výtahu je realizován v jedné ose, ze které bude využita informace o zrychlení. Polohu lze získat dvojitou integrací tohoto zrychlení podle času, neboť platí rovnice 5.1.

$$\ddot{z} = \frac{d\dot{z}}{dt} = \frac{d^2z}{dt^2} \tag{5.1}$$

Jelikož je nutné výpočet provádět v diskrétním časovém okamžiku, platí tato rovnice

$$\ddot{z} = \frac{\Delta\dot{z}}{\Delta t} \tag{5.2}$$

Výpočet aktuální rychlosti popisuje rovnice 5.3.

$$\dot{z}(t) = \dot{z}(t - \Delta t) + (\ddot{z}(t) \cdot \Delta t) \tag{5.3}$$

Kde $\dot{z}(t)$ je aktuální rychlost, $\dot{z}(t - \Delta t)$ je rychlost v předchozím časovém okamžiku, $\ddot{z}(t)$ je aktuální zrychlení a Δt je časová změna v průběhu jednoho kroku. Stejným způsobem lze z rychlosti určit ujetou vzdálenost.

Při znalosti ujeté vzdálenosti, které odpovídá jízdě o jedno podlaží, je možné určit aktuální podlaží, ve kterém se nachází výtah. Toto řešení však doprovází několik problémů. Jedná se především o offset a šum senzoru. Tyto problémy a možná řešení budou popsány v následující části.

5.2.2 DC bias

Toto řešení doprovází několik problémů. Měření je zatíženo šumem. Dalším důležitým faktorem je tíhové zrychlení, které působí v záporném směru osy z. Tíhová složka zůstává v průběhu měření stejná, avšak z důvodu nepřesnosti měření se po jejím odečtení v signálu objevuje zbytková stejnosměrná složka, označovaná jako *bias*. Tato stejnosměrná složka generuje při integraci přírůstky v poloze, které nejsou reálné. Tento jev je označován jako *drift*. Cílem je upravit signál zrychlení tak, aby byla v co nejvyšší míře tato stejnosměrná složka odstraněna. Toho lze dosáhnout kalibrací senzoru. Na základě kalibračního měření bude zjištěna střední hodnota, která bude z naměřeného signálu odečtena. Dalším možným řešením je filtrace signálu, kdy se ve frekvenční oblasti odstraní frekvence blízké nule, které odpovídají stejnosměrným složkám v signálu. Tato řešení lze aplikovat na naměřený signál, který je zpětně zpracován. Pro systémy, které jsou provozované v reálném čase, není možné tyto přístupy použít. S ohledem na tyto problémy musí být navrženo takové řešení, které bude vhodné pro systém využívaný pro aplikaci v reálném čase.

5.2.3 Návrh řešení

S ohledem na výše zmíněné informace je nutné vyvinout systém tak, aby bylo možné co nejvíce eliminovat stejnosměrnou složku v signálu při zachování požadavků na použití. Proto bylo navrženo řešení, které tyto vlivy úspěšně potlačuje. Návrh lze rozdělit do těchto částí.

- Kalibrace senzoru
- Kalibrace polohy
- Algoritmus eliminace biasu
- Výpočet aktuálního patra

Kalibrace senzoru

Výpočet střední hodnoty je realizován kalibrací senzoru, kdy je po určenou dobu měřeno zrychlení. Z těchto hodnot je vypočítán průměr a tato hodnota je pak odčítána od všech následujících hodnot měření.

Kalibrace polohy

Důležitou částí systému je přizpůsobení na daný výtah, ve kterém bude jízda realizována. Jedná s o maximální dosahovaná zrychlení, rychlosti a o vzdálenost, kterou ujede výtah při popojetí o jedno podlaží. Díky této vzdálenosti je možné určit aktuální podlaží z informace o poloze. Na začátku každé jízdy v dosud neznámém výtahu je nutné uskutečnit jízdu o jedno podlaží. V průběhu jízdy jsou ukládány hodnoty zrychlení v čase. Po zastavení jsou tato data vyhodnocena.

Vyhodnocení má dvě fáze. V první fázi je z uloženého zrychlení eliminován DC bias tak, že je vytvořena pásmová propust, která odstraní frekvence blízké 0 a dále frekvence vyšší jak 10 Hz. Touto úpravou je odstraněn bias spolu s vysokofrekvenčním šumem. Následně je ze signálu vypočtena rychlost. Z ní je zjištěna maximální hodnota rychlosti vel_{max} a zároveň maximální hodnota diference rychlosti Δvel_{max}.

Aby výpočet polohy korespondoval s další částí programu, tak je nutné postupovat totožně. Proto následuje druhá fáze, kdy jsou využity získané informace a na akcelerometrická data je aplikován algoritmus eliminace biasu, který je vysvětlený v sekci Algoritmus . Výstupem z algoritmu je informace o poloze, která byla uražena. Maximální hodnota polohy odpovídá vzdálenosti ujeté o jedno podlaží. Výstupem z této části je tedy maximální hodnota rychlosti vel_{max}, maximální hodnota diference rychlosti Δvel_{max} a vzdálenost, kterou ujede výtah při jízdě o jedno podlaží. Tyto hodnoty jsou vstupem pro další fázi.

Algoritmus eliminace biasu

Jak již bylo řečeno, částečné potlačení biasu a následných negativních vlivů je dosaženo kalibrací senzoru. Avšak kalibrace nedokáže bias odstranit kompletně. V průběhu měření je bias stále přítomný a tím pádem je v integraci kumulována chyba. Z toho důvodu byl navrhnut algoritmus pro úpravu rychlosti z integrace zrychlení.

Algoritmus využívá nastavení hraničních hodnot na základě Kalibrace polohy. V případě, že nejsou splněny definované podmínky, je hodnota rychlosti vždy nastavena na fixní hodnotu 0. K úspěšné funkčnosti je nutné dokázat definovat tyto důležité oblasti v signálu.

1) Rozjezd výtahu
2) Jízdu konstantní rychlostí
3) Zpomalování výtahu
4) Zastavení

V případě, že je detekována dostatečně strmá hrana v signálu, je zapnuta nebo případně vypnuta integrace. V závislosti na směru a velikosti aktuálního a předchozího stavu jsou detekována důležitá místa v signálu a díky této informaci je možné upravit rychlost tak, aby byl zcela odstraněn bias. Pseudo-kód tohoto prezentovaného algoritmu je na Obr. 5.2.

Inicializace všech hodnot je nastavena na 0. V *kroku 1* je vždy vypočítána rychlost $vel(t)$ a její diference Δvel na základě předchozí rychlosti $vel(t-1)$, aktuálního zrychlení $acc(t)$, předchozího zrychlení $acc(t-1)$ a na základě diference času Δt. V *kroku 2* je zjištěno, zda je zapnuta integrace, nastavena na hodnotu True. V případě, že je integrace zapnuta, tak je dále zjištěno, zda Δvel kleslo pod stanovenou hodnotu, viz. *krok 3*. Pokud ano, tak je nutné rozlišit, zda se výtah nachází v části konstantního zrychlení, *krok 4-6* nebo výtah zastavuje, *krok 7-9*. Na základě této informace je hodnota rychlosti zafixována, respektive nastavena na 0. Pokud Δvel nekleslo pod stanovenou hodnotu, tak se nic nemění a krok algoritmu je ukončen.

Pokud nebyla splněna podmínka v *kroku 2*, znamená to, že je integrace vypnuta, nastavena na hodnotu False a pokračuje se *krokem 12*. Dále je nutné zjistit, zda Δvel překročilo stanovenou hodnotu. V závislosti na podmínkách je detekováno kladné překročení, *krok 13* nebo záporné překročení *krok 15*. V obou případech je integrace nastavena na hodnotu True. V případě, že není splněna žádná z podmínek, je zachována předchozí hodnota rychlosti. Na závěr je vracen výstup v podobě aktualizované rychlosti $vel(t)$. Na základě výstupu je vypočítána poloha a aktuální podlaží.

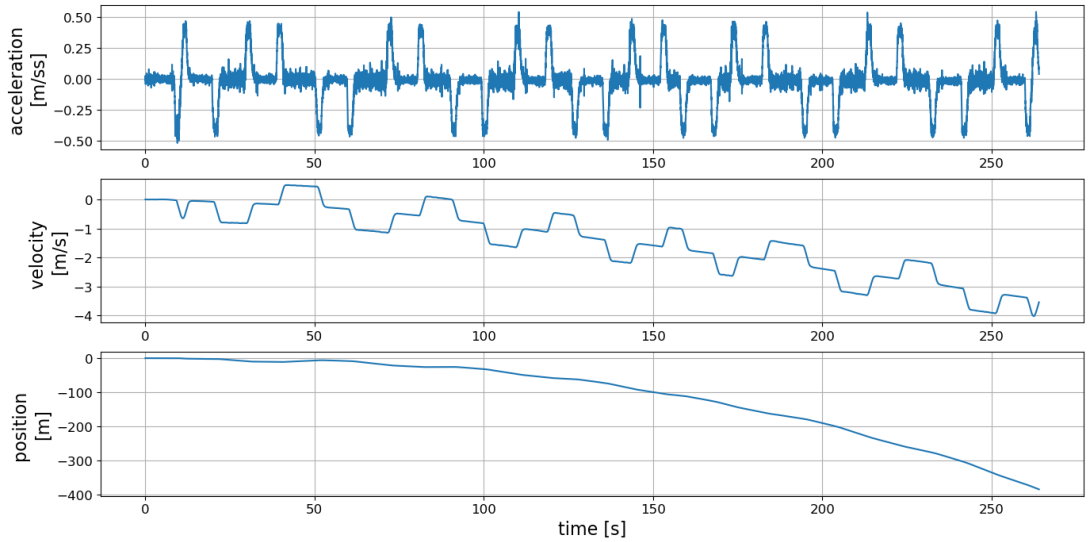
Algorithm 1 Pseudo-kód algoritmu eliminace biasu

$vel(t), \Delta vel = \text{getVelocity}(vel(t-1), acc(t), acc(t-1), \Delta t)$
if *integrate* is True **then**
    if $\Delta vel < max_{diff} * treshold_{low}$ **then**
        if $abs(vel) > 0.5 * vel_{max}$ **then**
            $integrate = \text{False}$
            $vel(t) = vel(t-1)$
        else
            $vel(t) = 0$
            $integrate = \text{False}$
    else
        pass
else
    if $\Delta vel >= max_{diff} * treshold_{pos}$ **then**
        $integrate = \text{True}$
    else if $abs(\Delta vel) >= max_{diff} * treshold_{neg}$ and $\Delta vel < 0$ **then**
        $integrate = \text{True}$
    else
        $vel(t) = vel(t-1)$
return $vel(t)$

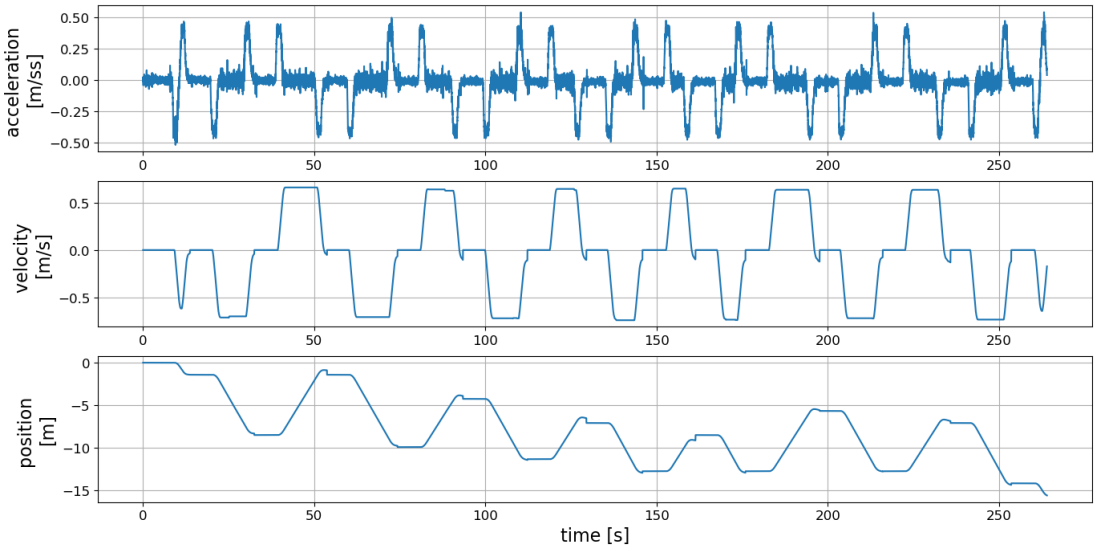
Obr. 5.2 – Pseudo-kód algoritmu eliminace biasu

5.2.4 Výstup

Výsledek bez použití algoritmu pro eliminaci biasu je na Obr. 5.3. Můžeme vidět, že výpočet polohy je velmi ovlivněn driftem a s tímto výsledkem není možné dále pracovat.



Obr. 5.3 – Výsledek bez úpravy rychlosti



Obr. 5.4 – Výsledek po použití algoritmu eliminace biasu

Výsledek při použití algoritmu je zobrazen na Obr. 5.4, kde je rychlost a poloha nezatížena driftem. Algoritmus byl otestován na 16 různých výtahových kabinách. Výsledky byly zaznamenány do tabulky. Na základě další využitelnosti algoritmu byla vyhodnocena přesnost relativní změny patra. Řádky tabulky popisují reálnou změnu patra. Sloupce tabulky popisují, jaký výsledek byl určen detekcí za předpokladu reálné změny. Tato tabulka se bude dále využívat při datové fúzi.

Tab. 5.2 – Matice záměn relativní změny podlaží

		Detekovaná změna podlaží														
	-	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Reálná změna podlaží	**0**	**60**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	**58**	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0
	3	0	0	0	**33**	3	0	0	0	0	0	0	0	0	0	0
	4	0	0	0	2	**45**	7	0	0	0	0	0	0	0	0	0
	5	0	0	0	0	2	**33**	12	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	2	**29**	9	0	0	0	0	0	0	0
	7	0	0	0	0	0	0	1	**16**	7	0	0	0	0	0	0
	8	0	0	0	0	0	0	0	0	**24**	8	1	0	0	0	0
	9	0	0	0	0	0	0	0	0	3	**10**	9	0	0	0	0
	10	0	0	0	0	0	0	0	0	0	1	**5**	2	0	0	0
	11	0	0	0	0	0	0	0	0	0	0	1	**5**	2	0	0
	12	0	0	0	0	0	0	0	0	0	0	0	1	**1**	2	0
	13	0	0	0	0	0	0	0	0	0	0	0	0	1	**1**	2
	14	0	0	0	0	0	0	0	0	0	0	0	0	1	1	**1**

V předchozím testu byla vyhodnocena absolutní hodnota relativní změny, avšak pro možné rozšíření robustnosti datové fúze byly provedeny testy s ohledem na kladné i záporné změny, které umožní určit nejen relativní posuv, ale i směr pohybu výtahu. Tyto výsledky zobrazuje tabulka A, která je součástí přílohy.

Z výsledů je patrné, že detekce není závislá na směru jízdy. Přesnosti kladných a záporných relativních změn vykazují podobný trend a rozložení.

5.3 Detekce podlaží na základě obrazových dat

Druhým přístupem pro úlohu detekce podlaží výtahu je algoritmus, který využívá obrazových dat z displejů. V této kapitole bude představeno řešení, které je vyvinuté na základě informací z rešeršní části.

5.3.1 Definice úlohy

Základní myšlenkou detekce podlaží na základě obrazových dat z displejů je využití číselné informace, která zobrazuje aktuální číslo podlaží. Cílem je klasifikovat tato obrazová data na základě zobrazovaného čísla do klasifikačních tříd. Klasifikace bude prováděna pomocí klasifikační CNN, jejíž model bude natrénován a otestován na vytvořené datové sadě, která je prezentována detailně v kapitole 4.1. V datové sadě se objevují pouze obrázky displejů vygenerované z anotovaných videí. Detekce displeje není součástí této práce, proto je vždy předpokládáno, že obrazovým vstupem je obrázek obsahující pouze informační displej.

5.3.2 TensorFlow 2 a Keras API

Nejprve je nutné určit, jaký nástroj bude k realizaci úlohy použit. Pro realizaci celé úlohy byl vybrán programovací jazyk Python, který díky různým modulům nabízí široké možnosti využití. V oblasti strojového učení je zde několik možných nástrojů. Jedná se o např. o knihovny TensorFlow a PyTorch.

Pro potřeby této práce byla vybrána platforma TensorFlow 2 s využitím Keras API, které slouží jako vysokoúrovňové API pro práci s touto platformou. Tato kombinace byla vybrána díky široké využitelnosti, detailní dokumentaci a intuitivní implementaci. Další výhodou při použití těchto nástrojů je vzájemná kompatibilita, která bude zaručena pro ostatní funkční celky práce. Z důvodu vzájemné závislostí jednotlivých balíčků byla po dohodě vybrána verze 2.3, která je kompatibilní s použitou verzí Pythonu 3.8. Pro dodržení kompatibility musí být použity verze programů, které je možno nalézt v příloze. V opačném případě nemůže být zaručena vzájemná kompatibilita a funkčnost. Výhodou je také dostupnost základních modelů předtrénovaných CNN, které lze v rámci aplikace přímo využívat. Aplikace obsahuje všechny funkce, potřebné pro vytvoření datové sady, augmentaci dat, práci s modelem, proces učení a validaci.

5.3.3 Preprocessing vstupních dat

První fází klasifikace obrazu je část, kdy jsou obrazová data zpracována do optimální podoby. Preprocessing se skládá ze změny velikosti při zachování poměru stran a z redukce šumu.

Knihovna, která je využívána při preprocessingu se nazývá OpenCV[16] a obsahuje funkce, které usnadňují implementaci algoritmů zpracování obrazu.

Změna velikosti

Pro potřeby práce je nutné upravit velikosti obrazu tak, aby odpovídala vstupu do neuronové sítě. Tato změna velikosti je uskutečněna tak, aby byl zachován poměr stran obrazu. Vstupem do neuronové sítě bude vždy čtvercový obraz. Proto je v první fázi původní obraz vložen do čtvercového obrazu, který má velikost hrany stejnou, jako je nejdelší hrana původního obrazu. Následně je velikost nového obrazu změněna tak, aby odpovídala požadované velikosti.

Redukce šumu

Poslední fází je redukce šumu v obrazu. K tomu je využita funkce GaussianBlur. Jedná se konvoluci obrazu s Gaussovým filtrem. Tato operace vede k redukci Gaussova šumu v obrazu a má za cíl zlepšit úspěšnost klasifikace.

5.3.4 Postup učení sítě EfficientNet-B0

V rešeršní části byla vybrána síť EfficientNet. Konkrétně verze B0, která je první verzí z rodiny těchto CNN. Síť má 5,3 mil. parametrů a definovaný vstup o velikosti 224x224 pixelů.

Postup tréninku probíhal následovně.

1) Načtení datové sady a augmentace dat
2) Načtení a úprava modelu
3) Fáze učení

Následně byl natrénovaný model otestován na testovacích datech.

Načtení datové sady a augmentace dat

Nejprve je nutné importovat datovou sadu do podoby, která je definovaná platformou TF2. K tomu slouží třída *ImageDataGenerator*, která v sobě zahrnuje několik metod a funkcí.

Jednou z funkcí je možnost augmentace dat. Augmentace dat je využívána pro zvětšení trénovací datové sady, kdy jsou původní data rozšířena o nově vytvořené modifikace. Příkladem augmentace může být náhodné natáčení obrazu nebo náhodný zoom. Datová sada s malým počtem prvků je velmi náchylná na *přetrénování*.

Při tréninku bylo využito náhodné rotace o max. 8 stupňů, náhodný zoom v intervalu 95 až 105 % a shear[17] 0.15.

Další metodou je generování datové sady přímo ze souboru. Formát souboru musí být takový, že každé klasifikační třídě odpovídá jedna složka a název této složky odpovídá štítkům označujícím data příslušné třídy. Vstupní parametry metody jsou uvedené v tabulce Tab. 5.3.

[16] Open Source Computer Vision
[17] Fixace v jedné ose, natočení v ose druhé

Tab. 5.3 – Vstupní parametry funkce generování datové sady

Název vstupu	Popis
Color_mode	Definuje, zda se jedná o barevný nebo černobílý vstup.
File_path	Cesta k souboru s datové sadě.
Target_size	Velikost vstupních dat.
Batch_size	Velikost batch[18].
Class_mode	Mód třídy.
Subset	Označení validační/trénovací datové sady.

Pro trénování byla využita datová sada prezentovaná v podkapitole 4.1.2. Byla rozdělena v poměru 80/20. Větší část je použita pro trénování, menší část je využita pro validaci.

Načtení a úprava modelu

Vybraný model je součástí prostředí *Keras* uložený v části *application*, proto je jeho načtení definované jako funkce s názvem EfficientNetB0. Ve funkci je možné definovat, zda se model načte celý nebo bez horní (klasifikační) vrstvy. Dále je definovaná velikost vstupního tensoru a jsou načtené váhy, které jsou opět součástí *Keras*.

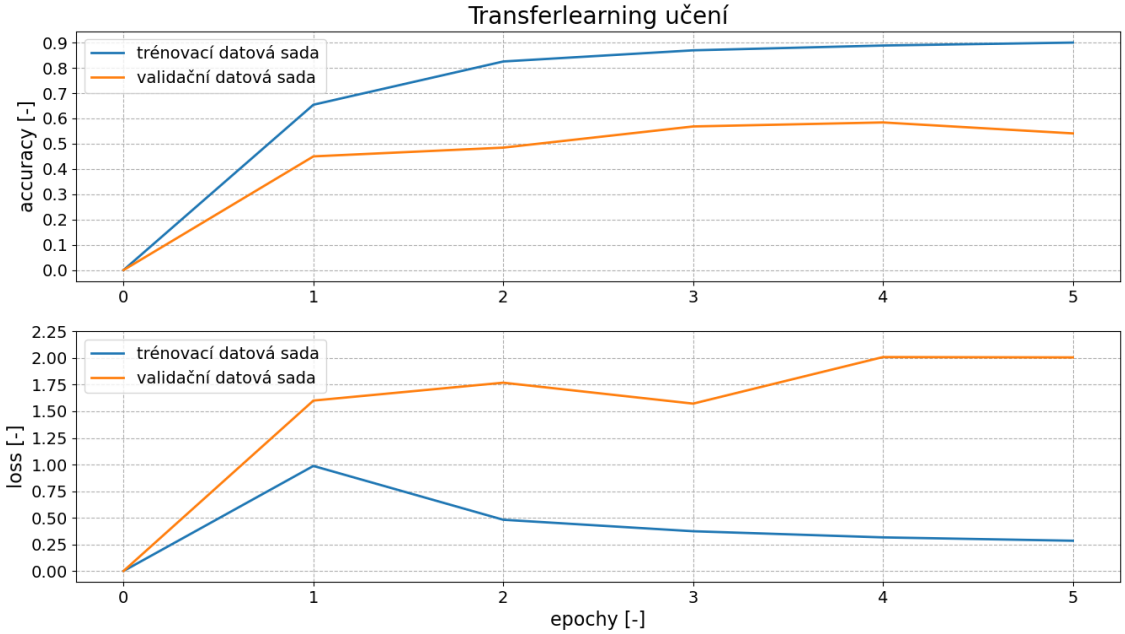
Následně je nutné zakázat aktualizaci vah modelu. Tím je určeno, že se skryté konvoluční vrstvy nebudou měnit a bude tak zachována jejich informace. Poté je nutné vytvořit výstupní část modelu, která nebyla načtena. Konkrétně byly přidány 4 plně propojené vrstvy. Poslední vrstva má nastavenu aktivační funkci softmax.

Poslední fází je kompilace modelu. Tak je model připraven na proces učení.

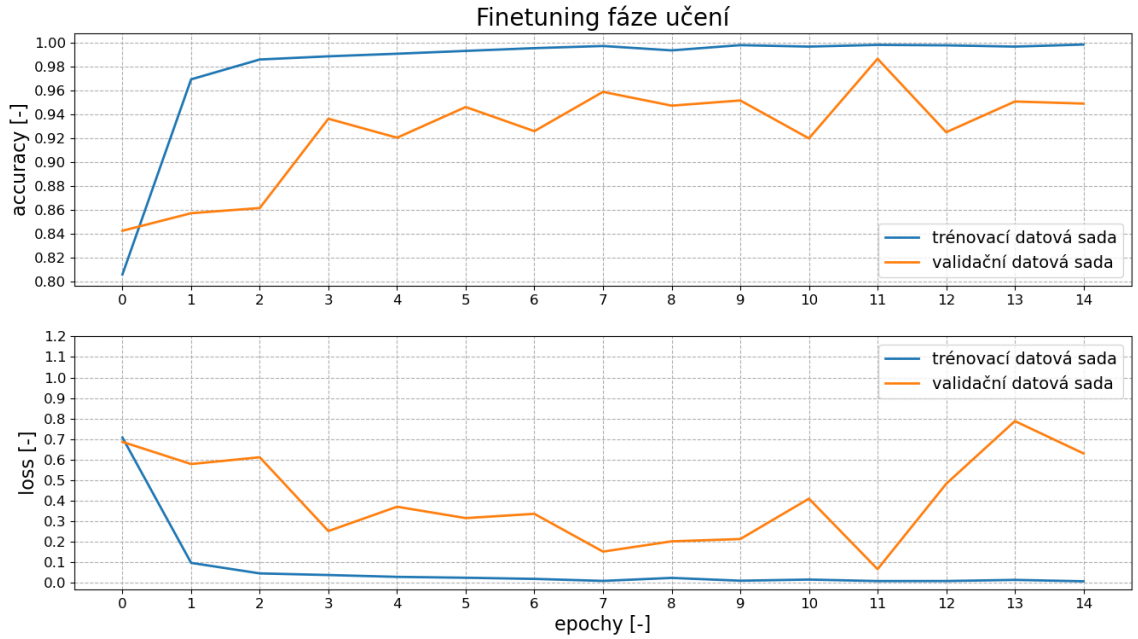
Učení

Učení má dvě části. První částí je natrénování klasifikačních vrstev. Pro učení je využita metoda *fit*. Jejím vstupem je trénovací datová sada, počet epoch a další volitelné parametry. Počet epoch byl nastaven na 5. Jako volitelný parametr bylo zvoleno přidání validační datové sady. Jedná se o funkci, kdy je po každé epoše otestována přesnost modelu. Zde je možné v průběhu učení zjistit, jaké výsledky poskytuje model a zda se nedostal do fáze *přetrénování*. Druhou částí je finetuning učení. Pro dvacet posledních vrstev modelu byla povolena aktualizace vah. Tyto vrstvy je proto možné trénovat. Dalším důležitým krokem je nastavení koeficientu učení, které je nutné nastavit na velmi malou hodnotu.

[18] definuje velikost intervalu aktualizace vah (např. po každém 20. je provedeno zpětné šíření chyby)



Obr. 5.5 – Průběh transfer learning fáze



Obr. 5.6 – Průběh finetuning fáze

Z průběhů učení je možné vidět, že v první fázi na Obr. 5.5 se přesnost na trénovacích datech velmi rychle blíží k 90 %, avšak přesnost na validační datové sadě je maximálně 50 %. Tento velký rozdíl značí, že se síť rychle adaptuje pouze na trénovací datovou sadu a nastává *overfitting*. Proto je nutné postoupit k druhé fázi učení. V této fázi, která je zobrazena na Obr. 5.6 se úspěšnosti na testovací, a především na validační datové sadě zvyšují. Nejlepších výsledků je dosaženo v epoše 11, kde je přesnost na validační datové sadě se blíží 99 % a

hodnota *loss function* klesá pod 0,1. Síť je tedy připravena na otestování. Výsledky jsou zobrazené v kapitole 5.3.5.

5.3.5 Výsledky

Testování probíhá tak, že je predikce prováděna na datech, které nebyly použité v trénovací datové sadě. Výsledky jsou prezentované ve formátu *matice záměn.* Jedná se o zápis, kdy řádky představují skutečnou klasifikační třídu a sloupce odhad klasifikátoru. Na diagonále se tedy objevují počty prvků, které klasifikátor zařadil správně.

Tab. 5.4 – Matice záměn, klasifikace obrazových dat

	Odhad klasifikační třídy															
Skutečná klasifikační třída	**-**	**-2**	**-1**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
	-2	**35**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-1	0	**179**	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	**496**	1	0	0	0	0	0	0	2	0	0	0	0
	1	0	2	24	**1221**	0	1	4	0	0	1	0	0	14	4	0
	2	0	0	2	8	**1946**	0	1	17	9	0	0	2	0	4	0
	3	0	0	0	0	0	**1022**	0	2	1	0	0	0	0	0	0
	4	0	0	1	14	0	0	**901**	5	0	0	0	1	0	0	0
	5	0	0	7	0	0	3	0	**1074**	47	0	0	5	0	0	0
	6	0	0	1	1	0	4	0	1	**1059**	0	0	0	0	0	0
	7	0	0	1	11	0	8	2	0	0	**853**	0	0	0	0	0
	8	0	0	0	0	0	0	0	0	0	0	**212**	0	0	0	0
	9	0	0	0	0	0	0	0	0	0	0	0	**197**	0	0	0
	10	0	0	0	0	0	0	0	0	0	0	0	0	**105**	0	0
	11	0	0	0	0	0	0	0	0	0	0	0	0	1	**71**	0
	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**70**

Z tabulky je patrné, že chybovost je z velké části mezi visuálně podobnými čísly. Jedná se např. o dvojice 2,5; 1,7; 1,10; 0,8; 1,11.

5.4 Fúze dat

Posledním důležitým celkem je fúze výsledků výše zmíněných částí do jednoho s cílem zvýšit robustnost výsledného systému.

5.4.1 Definice vstupů

Definice vstupů má dvě roviny. První rovinou jsou vstupy, které vycházejí z úloh detekce patra na základě akcelerometrických a obrazových dat. Jedná se tedy o dvě konkrétní číselné hodnoty. První popisuje *změnu podlaží*, která byla detekovaná mezi rozjezdem a zastavením výtahu, druhá popisuje *číslo aktuálního podlaží*, které je zobrazeno na vnitřním informačním displeji výtahu.

Druhou zmíněnou rovinou jsou informace o podmíněných pravděpodobnostech, které jsou stěžejní pro správnou funkci datové fúze. Jedná se o data, která hodnotí úspěšnost jednotlivých detekcí. Tabulky jsou vytvořené na základě jejich testování. Jedná se o data, která přímo vycházejí z Tab. 5.2, která popisuje úspěšnost detekce na základě akcelerometrických dat a Tab. 5.4, která popisuje úspěšnost detekce na základě obrazových dat, tedy na základě úspěšnosti predikce CNN. Čísla v tabulkách jsou normalizované na hodnotu pravděpodobnosti mezi 0–1. Z důvodu zajištění numerické stability výpočtu je ke každé hodnotě Tab. 5.5 přičtena velmi malá nenulová hodnota, která tuto stabilitu zajistí. V Tab. 5.6 z důvodu vyššího počtu vzorků je provedena inicializace hodnotou 1. Tím je opět zajištěna stabilita řešení a zároveň není nijak narušena korektnost výpočtu.

Tab. 5.5 – Přesnost algoritmu pro výpočet relativní změny podlaží

	P(detekovaná změna podlaží | skutečná změna podlaží)															
Skutečná změna podlaží	-	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	1	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	2	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	3	0.00	0.00	0.00	0.92	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	4	0.00	0.00	0.00	0.04	0.83	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	5	0.00	0.00	0.00	0.00	0.04	0.70	0.26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.05	0.73	0.23	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	7	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.67	0.29	0.00	0.00	0.00	0.00	0.00	0.00
	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.73	0.24	0.03	0.00	0.00	0.00	0.00
	9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.14	0.45	0.41	0.00	0.00	0.00	0.00
	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.63	0.25	0.00	0.00	0.00
	11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.63	0.25	0.00	0.00
	12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.25	0.50	0.00
	13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.25	0.50
	14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.33	0.33

Tab. 5.6 – Přesnost klasifikace obrazových dat

	P(detekované podlaží |skutečné podlaží)															
Skutečné podlaží	-	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12
	-2	0.720	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020
	-1	0.005	0.928	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
	0	0.002	0.002	0.967	0.004	0.002	0.002	0.002	0.002	0.002	0.002	0.006	0.002	0.002	0.002	0.002
	1	0.001	0.002	0.019	0.950	0.001	0.002	0.004	0.001	0.001	0.002	0.001	0.001	0.012	0.004	0.001
	2	0.000	0.000	0.001	0.004	0.972	0.000	0.001	0.009	0.005	0.000	0.000	0.001	0.000	0.002	0.000
	3	0.001	0.001	0.001	0.001	0.001	0.984	0.001	0.003	0.002	0.001	0.001	0.001	0.001	0.001	0.001
	4	0.001	0.001	0.002	0.016	0.001	0.001	0.963	0.006	0.001	0.001	0.001	0.002	0.001	0.001	0.001
	5	0.001	0.001	0.007	0.001	0.001	0.003	0.001	0.934	0.042	0.001	0.001	0.005	0.001	0.001	0.001
	6	0.001	0.001	0.002	0.002	0.001	0.005	0.001	0.002	0.981	0.001	0.001	0.001	0.001	0.001	0.001
	7	0.001	0.001	0.002	0.013	0.001	0.010	0.003	0.001	0.001	0.960	0.001	0.001	0.001	0.001	0.001
	8	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.938	0.004	0.004	0.004	0.004
	9	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.934	0.005	0.005	0.005
	10	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.883	0.008	0.008
	11	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.023	0.828	0.011
	12	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.835

5.4.2 Implementace Bayesova filtru

Obecný algoritmus Bayesova filtru je představen v kapitole 2.2.1. Na základě zmíněného popisu byl algoritmus implementován na daný problém. S odkazem na výše zmíněnou kapitolu budou jednotlivé prvky značené shodně.

Tvorba Bayesova filtru

Změna podlaží vystupující z detekce pater na základě akcelerometrických dat bude značena jako u_t a číslo podlaží, které je výsledkem klasifikace obrazu displeje snímaného pomocí kamery bude značené jako z_t. Jelikož se jedná o rekurzivní algoritmus, vstupem je také výsledek z předchozího kroku značený jako $bel(x_{t-1})$. Proměnná x_t značí aktuální podlaží, x_{t-1} značí podlaží v předchozím kroku. Výsledná důvěra $bel(x_t)$ tedy určuje pravděpodobnost s jakou je detekované příslušné podlaží. To s nejvyšší hodnotou důvěry je poté považováno za skutečně dosažené podlaží.

Prvním krokem algoritmu je predikce $\overline{bel}(x_t)$, prvotní odhad na základě akcelerometrických dat. Mějme modelový příklad, kdy je počítána hodnota predikce pro $x_t = 2$. Tedy pravděpodobnost, že se robot nachází v podlaží číslo 2. Budova má N pater. Program akcelerometru hlásí změnu podlaží $u_t = 1$. Výtah tedy dle měření popojel o 1 podlaží. Pro výpočet je znám předchozí krok $bel(x_{t-1})$. Výpočet predikce pro podlaží 2 je popsán rovnicí 5.4.

$$\begin{gathered}\overline{bel}(x_t = 2) = \\ P(x_t = 2|u_t = 1, x_{t-1} = 0) * bel(x_{t-1} = 0) \\ + \\ P(x_t = 2|u_t = 1, x_{t-1} = 1) * bel(x_{t-1} = 1) \\ + \\ \dots \\ + \\ P(x_t = 2|u_t = 1, x_{t-1} = N) * bel(x_{t-1} = N)\end{gathered} \quad 5.4$$

Jsou tedy vypočítány všechny možnosti. Druhou fází je korekce. Zde je využita informace z kamery z_t. Pokračujme v modelovém příkladu a předpokládejme, že $z_t = 2$, tedy kamera hlásí, že se robot nachází v podlaží 2. Výstupem je hodnota pravděpodobnosti příslušeného patra.

$$bel(x_t = 2) = \eta P(z_t = 2|x_t = 2) * \overline{bel}(x_t = 0) \quad 5.5$$

Tímto způsobem je vypočítáno rozdělení pravděpodobnosti přes každé podlaží od 1 po N. Nejvyšší hodnotu pravděpodobnosti má podlaží, které je určené jako aktuální dosažené.

Zásadním prvkem v rovnicích jsou hodnoty podmíněných pravděpodobností $P(x_t|u_t,\ x_{t-1})$ a $P(z_t|x_t)$, které musí být známy. Tyto hodnoty je možné získat z tabulek podmíněných pravděpodobností definovaných v předchozí části. První zmíněnou podmíněnou pravděpodobnost je ovšem nutné upravit do podoby, která odpovídá tabulce. Využijeme tedy znalost teorie podmíněných pravděpodobností a provede úpravu.

$$P(x_t|u_t,\ x_{t-1}) = \mathrm{P}(\mathrm{u_t}|\ x_{t-1},\ x_t\ )P(x_{t-1}|x_t)P(x_t) \quad 5.6$$

Členy $P(x_{t-1}|x_t)$ a $P(x_t)$ můžeme zanedbat z důvodu, že jejich rozdělení pravděpodobnosti je rovnoměrné. To znamená, že na výsledek nemá vliv. Proto předpokládáme tento vztah.

$$P(x_t|u_t,\ x_{t-1}) \approx \mathrm{P}(\mathrm{u_t}|\ x_{t-1}, x_t) \quad 5.7$$

Poslední úprava je již formální. Jedná se o to, že je hledána pravděpodobnost určené relativní změny patra $\mathrm{u_t}$ za předpokladu skutečné změny patra. Skutečná změna patra je reprezentována členy $x_t,\ x_{t-1}$ tak, že odpovídá jejich rozdílu. Výsledkem je tedy rovnice 5.8, která koresponduje s tabulkou Tab. 5.5.

$$\mathrm{P}(\mathrm{u_t}|\ x_{t-1},\ x_t) =\ \mathrm{P}(\mathrm{u_t}|(x_{t-1} - x_t)) \quad 5.8$$

Takto upravená pravděpodobnost reprezentuje důvěru ve výsledek algoritmu pro výpočet relativní změny podlaží.

Pravděpodobnost $P(z_t|x_t)$ nevyžaduje žádnou úpravu, protože přesně koresponduje s Tab. 5.6. Jedná se tedy o pravděpodobnost určeného podlaží za předpokladu reálného podlaží. Tato

pravděpodobnost reprezentuje důvěru ve výsledek klasifikace obrazových dat a určení aktuálního podlaží.

Popis kódu

Na základě tohoto konceptu bylo naprogramováno řešení.

```
def countProbability(acc_diff_floor, camera_floor, floor_count):

  bel_new = []

  for i in range(floor_count):
    # bel prediction initialization
    bel_pred = 0

    # ====== PREDICTION ======= #
    for j in range(floor_count):
      bel_pred += acc_probab_table[(abs(acc_diff_floor)), abs(j-i)]*bel[j]

    # ==== CORECTION ====== #
    bel_new.append(camera_probab_table[camera_floor, i]*bel_pred)

  # normalization of belief
  suma = sum(np.copy(bel_new))
  for i in range(len(bel)):
    bel_new[i] /= suma

  return bel_new
```

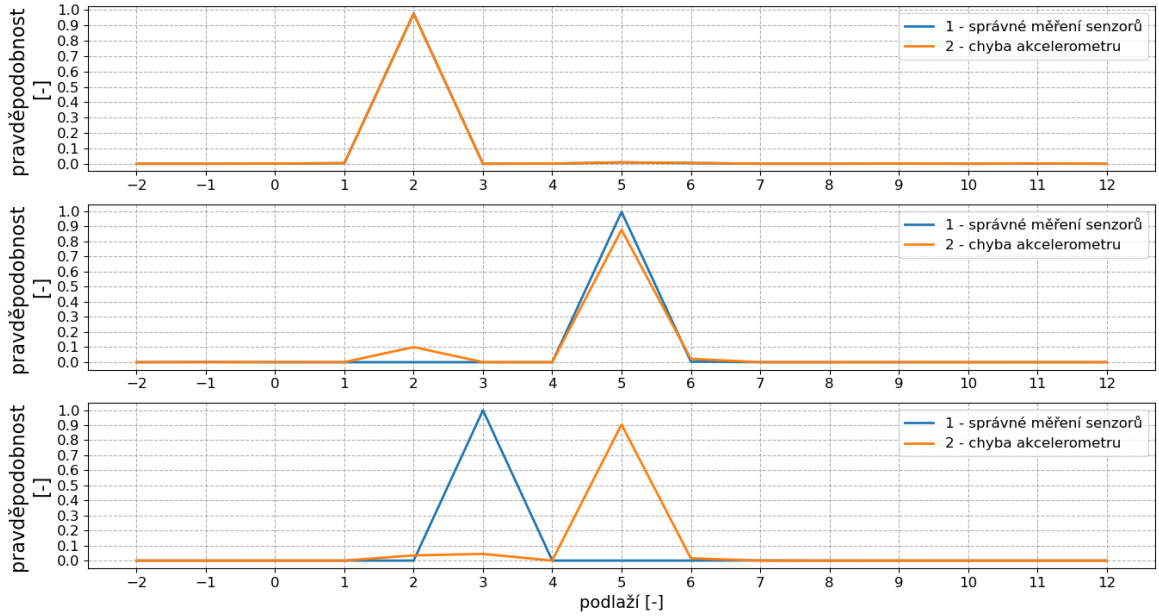
Vstupem do funkce *countProbability* je

- *acc_diff_floor* – označuje změnu podlaží
- *camera_floor* – označuje aktuální podlaží
- *floor_count* – označuje maximální počet pater, které je schopný systém detekovat

Seznam *bel_pred* obsahuje výsledky predikce, seznam *bel_new* obsahuje výsledek korekce, který je poté normalizován.

5.4.3 Ověření na simulovaných datech

Datová fúze byla nejprve ověřena při simulované jízdě výtahem. Nejprve byl definován průběh jízdy. Sekvence podlaží byla určena jako 2, 5, 3. Výsledek kamery byl vždy nastaven tak, aby odpovídal dané sekvenci. Výsledek akcelerometru byl při první iteraci nastaven také tak, aby odpovídal dané sekvenci, viz. křivka 1 v grafech na Obr. 5.7 V druhé iteraci byl pro srovnání jeho výsledek nastaven na 0. Změna patra tedy byla v průběhu celé jízdy nastavena chybně, viz. křivka 2. Výsledky pravděpodobnosti pro celý simulovaný průběh jsou zobrazeny na již zmíněném Obr. 5.7.



Obr. 5.7 – Výsledek datové fúze simulované jízdy výtahem

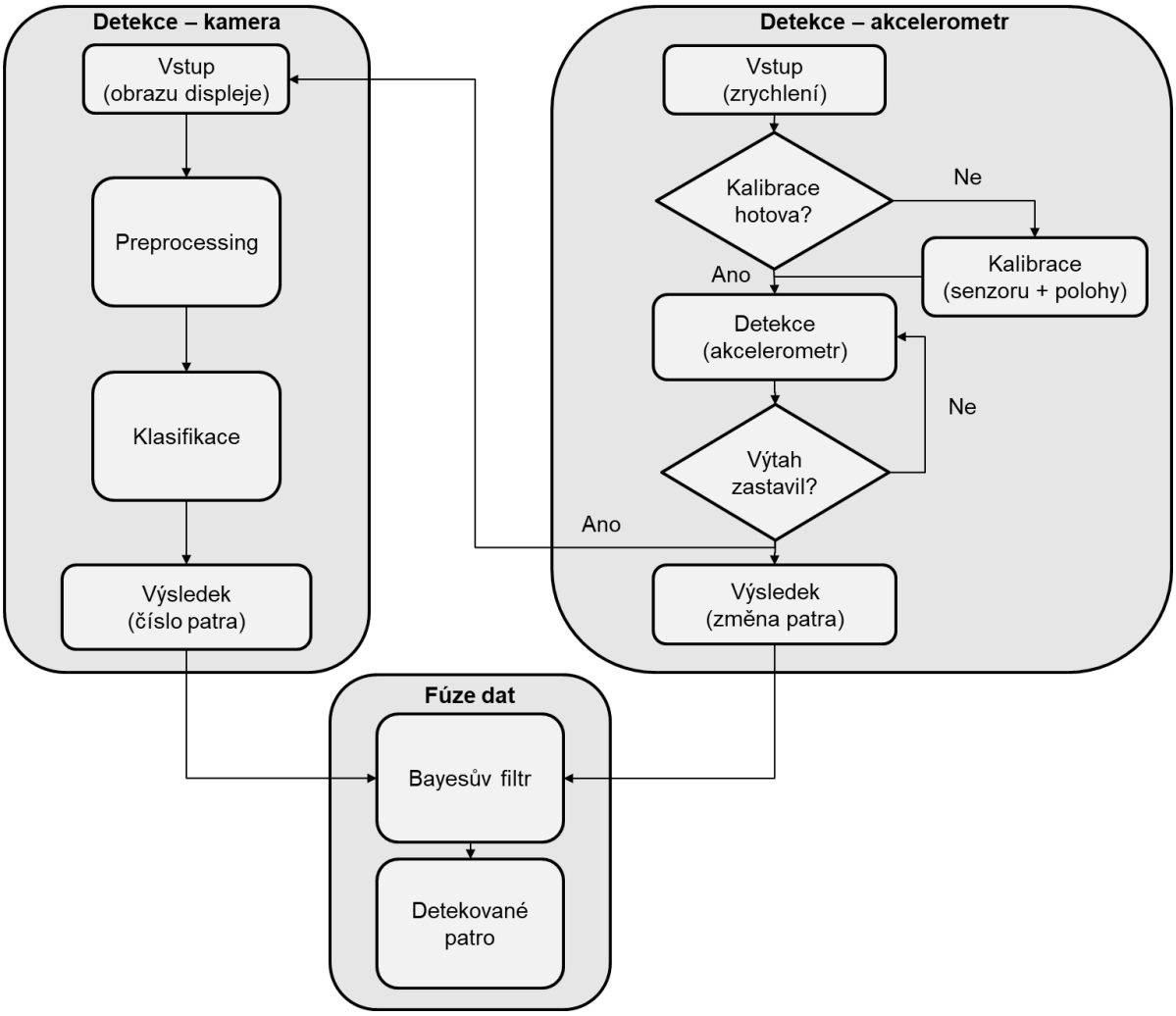
Z výsledků je možné vidět, že při prvním zastavení jsou obě křivky téměř shodné. Jedná se o to, že pro první zastavení je důvěra v předchozí podlaží definována rovnoměrným rozdělením pravděpodobnosti, tím pádem vliv chyby akcelerometru není viditelný. Obě křivky tedy správně označují jako výsledek podlaží č. 2.

Druhé zastavení je v podlaží č. 5. První křivka určuje správně tento výsledek. Křivka 2 určuje také jako výsledek podlaží č. 5, avšak výsledná pravděpodobnost je rozdělena mezi podlaží 2 a 5, což odpovídá předpokladu.

Poslední zastavení je v podlaží č. 3. Zde opět 1. křivka udává správný výsledek. Křivka 2 určuje jako výsledek podlaží č. 5. Pravděpodobnost je rozdělena téměř rovnoměrně mezi podlaží č. 3 a č. 5. Je možné vidět vyšší vliv chyby akcelerometru, který udává 0 změnu patra. Dle těchto výsledků je možno považovat chování datové fúze za korektní.

6 ALGORITMUS METODY DETEKCE AKTUÁLNÍHO PODLAŽÍ

Poslední fází vývoje programu je implementace všech jednotlivých částí do jednoho funkčního celku. Celý program je naprogramován v jazyce Python. K vytvoření byla použita technika OOP (objektově orientované programování). Jednotlivé důležité celky jsou definované jako objekty. Na základě tohoto přístupu je výsledný kód více přehledný a jednotlivé funkční celky jsou od sebe jasně odděleny. Základní schéma celého algoritmu je na obrázku Obr. 6.1.



Obr. 6.1 – Schéma algoritmu

První částí programu je detekce založená na měření akcelerometru popsaná v kapitole 5.2. Tato část má za cíl určit okamžik zastavení výtahu a jejím výstupem je změna podlaží mezi začátkem a koncem jízdy. V okamžiku, kdy je detekované zastavení výtahu, je načten obrázek displeje z kamery. Na tomto obrázku je provedena klasifikace a je určeno číslo na displeji, které odpovídá aktuálnímu podlaží. Tuto část popisuje kapitola 5.3. Výsledky jsou vstupem pro datovou fúzi, která je popsána v kapitole 5.4. Zde jsou výsledky spojeny do jednoho a

výstupem je číslo aktuálního podlaží, kterému odpovídá nejvyšší vypočtená hodnota pravděpodobnosti.

V následujících kapitolách bude detailněji popsáno celé fungování s odkazem na kód programu.

6.1 Vstupní parametry

Nastavení programu je prováděno přes vstupní parametry. Mezi nejdůležitější patří.

Tab. 6.1 – Důležité vstupní parametry aplikace

Název parametru	Popis
detection_mode	Definuje, zda bude aplikace v online nebo v offline módu
image_input_mode	Definuje, z jakého zdroje jsou načítány obrazová data.
best_weights_path	Cesta k nejlepšímu nastavení modelu CNN
floor_labels	List s pořadím a názvem všech možných podlaží od nejnižší po nejvyšší číslo

Dále je definovaná relativní cesta k hlavnímu adresáři, aby v případě využití na jiné platformě, či systému byla zaručena funkčnost odkazů k souborům. Jako další je definován COM port, přes který se zajišťuje komunikace senzoru zrychlení. Dále definice kamerového zařízení. V případě, že má systém více kamer, musí být definované, ze které jsou přijímána data.

6.2 Aplikace

Jak již bylo zmíněno pro každý funkční celek je deklarována speciální třída. Hlavním scriptem pro spuštění programu je App.py.

Tab. 6.2 – Důležité objekty použité v aplikaci

Název objektu	Umístění třídy objektu
floorEstimation	src/main.py
loadData	src/input_feed/InputFeed.py
accEstimateFloor	src/core/accelerometer_estimation/acc_estimate_floor.py
cameraEstimateFloor	src/core/camera_estimation/camera_estimate_floor.py
bayesFilter	src/bayes_filter/bayesFilter.py

V Tab. 6.2 můžeme vidět tyto objekty s názvem a umístěním jejich tříd. Jedná se o takovou strukturu, kdy ve třídě *floorEstimation* jsou v konstruktoru vytvořeny objekty ostatních tříd. To znamená, že při inicializaci tohoto objektu jsou ostatní objekty inicializované také. Každý objekt má metodu *spin()*. Jedná se o funkce, které jsou volané v hlavní smyčce programu. Struktura je tedy naprogramována tak, aby bylo možné chod programu provozovat v online (využití dat v reálném čase) nebo offline (s využitím předem naměřených dat) režimu. Podle režimu nastavení, jsou vybrány zdroje, odkud jsou data načítána. V následující části budou zmíněné části programu blíže popsány.

6.2.1 Načítání akcelerometrických a obrazových dat

Jedná se o objekt *loadData*, který slouží k načítání dat. Objekt dále zajišťuje kontrolu načtených dat. Na základě vstupního parametru *detection_mode,* je vybraný zdroj. Pro načítání akcelerometrických dat a dat obrazových jsou vytvořeny separátní třídy, jejichž objekty se inicializují v konstruktoru popisovaného objektu. Jedná se o objekty *AccDataSource a CameraDataSource*.

V případě, že je zvolena online detekce, je načítán obraz z kamery nebo z detekce ovládacích prvků a zrychlení z akcelerometru v reálném čase. V případě, že je zvolena offline detekce, je načítán obraz z videa nebo ze souboru. Zrychlení je načítáno ze souboru s naměřenými daty.

Pro všechny uvedené případy jsou vytvořeny speciální třídy děděním z abstraktních tříd. Díky tomu je možné inicializovat objekty se stejným názvem, stejnými metodami, které však mají rozdílnou funkci, s ohledem na výše zmíněné požadavky.

Příkladem může být objekt *AccDataSource*. Existují dva různé případy.

1) Online detekce

Objekt je inicializován z třídy AccRealTimeFeed, která je vytvořena děděním z abstraktní třídy AccFeed. Hlavní metodou je get_acceleration, která při volání funkce dává jako výstup aktuální zrychlení. V tomto případě zrychlení ze senzoru v reálném čase.

2) Offline detekce

Pokud je zvolena offline detekce, je objekt *AccDataSource* inicializován z třídy CsvFileFeed, která je opět vytvořena děděním z abstraktní třídy AccFeed. Opět je zde metoda get_acceleration, která má na výstupu zrychlení. Toto zrychlení je však načítáno ze souboru z měření.

Z tohoto příkladu je zřejmé, jaké výhody tento přístup má. Jedná se především o možnost používat stejný objekt, který je dle potřeby inicializován z funkčně odlišných tříd. Další možnou výhodou je jednoduchá implementace nové funkcionality a implementace změn v programu.

6.2.2 Implementace Detekce dle akcelerometrických dat

Objekt *accEstimateFloor* v sobě obsahuje řešení detekce podlaží na základě akcelerometrických dat. Detailní popis tohoto řešení je v kapitole 5.2. Z toho důvodu zde bude zmíněna základní logika v návaznosti na celý systém. Metoda, která je volána v hlavní smyčce programu je spin(). Jedná se o funkci, která v sobě integruje všechny prvky detekce. Je založena na logice stavového automatu. Program má 2 stavy.

- Kalibrace polohy („ride_calibration“); přizpůsobení na daný výtah,
- Detekce patra („estimate_floor“); výpočet změny podlaží v průběhu jízdy.

Vstupem je aktuální zrychlení, které je načteno již popsaným objektem *loadData*. Výstupem je aktuální změna podlaží v každém kroku. Stěžejní částí je určení zastavení výtahové kabiny. V případě, že výtah zastavil, tak je proměnná *elevators_ride_stop* nastavena na hodnotu 1 (true). Tato informace je využita v další části. Proměnná je na začátku každé iteraci automaticky nastavena zpět na 0 (false). V případě, že je detekováno zastavení výtahové kabiny, je

vypočtena změna podlaží mezi aktuální a startovací polohou. Tento výstup vstupuje do datové fúze.

6.2.3 Implementace Detekce podlaží dle obrazových dat

Jedná se o objekt *cameraEstimateFloor*. Tento objekt v sobě zahrnuje všechny části, které zajišťují klasifikaci dat z displeje. Vstupem je obrazový záznam displeje výtahu, jehož načtení opět zajišťuje objekt *loadData*. Při inicializaci byl načten model CNN sítě EfficientNetB0 a váhy, které byly získány ve fázi učení popsané v kapitole 5.3. Načítají se takové váhy, s kterými model dosahuje nejvyšší přesnosti na validačním datové sadě. Dále je definován list štítků, díky kterému je možné přiřadit název třídy k příslušné klasifikaci. Jedná se o sekvenci čísel podlaží ve vzestupném pořadu.

Hlavní metodou je opět metoda spin(), která je volána v hlavní smyčce programu. Je provedena právě tehdy, když je proměnná *elevators_ride_stop* nastavena na hodnotu 1. Jedná se o parametr objektu *accEstimateFloor* a podává informaci o zastavení výtahové kabiny. V případě, že je podmínka splněna, jsou provedeny jednotlivé části metody spin.

Je načten aktuální obraz displeje a v první fázi je provedeno jeho předzpracování. Upravený obraz poté vstupuje do klasifikátoru, který provede klasifikaci. Výsledkem je normovaný seznam skóre jistoty všech tříd daného vstupního obrazu. Na základě znalosti listu štítků je možné každé klasifikační třídě přiřadit její název.

Třída s nejvyšší hodnotou skóre určuje výsledek. V případě, že hodnota skóre je nižší než 80%, jedná se dle experimentálního ověření o chybnou klasifikaci. V tom případě není výsledek považován za platný.

Výstupem je název třídy, tzn. číslo aktuálního podlaží. Tento výsledek je vstupem pro datovou fúzi.

6.2.4 Implementace datové fúze

Poslední fází v aplikaci je fúze dat. Tato část je implementována v objektu *bayesFilter*. Datová fúze je realizována konceptem Bayesova filtru. Teoretický rámec je popsaný v kapitole 2.2.1, návrh a teoretické ověření je v popsané v kapitole 5.4. Z tohoto důvodu zde bude popis uveden v kontextu celkové aplikace.

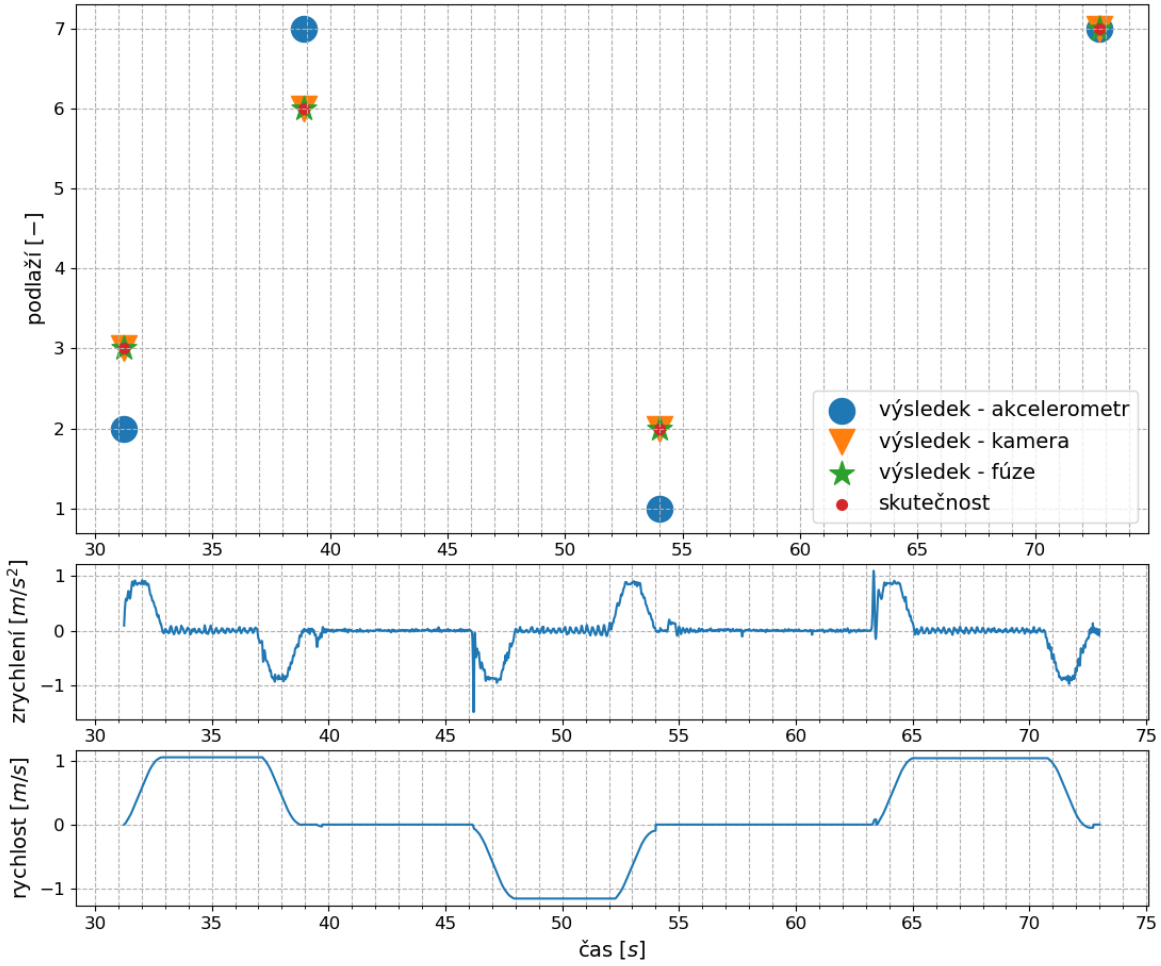
Vstupem jsou data z předchozích částí aplikace. Jedná se o výstupy z objektu *accEstimateFloor,* jehož výstupem je změna podlaží mezi startovací a aktuální polohou uložená do proměnné *acc_diff_floor*. Dále se jedná o výstup z objektu *cameraEstimateFloor*, jehož výstupem je číslo aktuálního podlaží uložené do proměnné *camer_floor*. Pro zaručení správné funkcionality musí být známy tabulky podmíněných pravděpodobností, které byly vytvořeny na základě vyhodnocení úspěšnosti zmíněných programů. Jedná se o Tab. 5.5, která je do programu načtena pod názvem *acc_probab_table* a Tab. 5.6, která je načtena pod názvem *camera_probab_table*.

Všechny části jsou integrovány do metody spin, která je spouštěna v hlavní smyčce programu. Nejprve jsou načteny příslušné tabulky, dále je na základě vstupů provedena datová fúze. Poté je proveden odhad na základě informace o změně podlaží *acc_diff_floor* v kombinaci s tabulkou podmíněných pravděpodobností *acc_probab_table.* Následně je provedena korekce na základě informaci o aktuálním podlaží *camer_floor* v kombinaci s tabulkou podmíněných

pravděpodobností *camera_probab_table*. Výstupem je normalizovaný seznam hodnot důvěry každého podlaží. To s nejvyšší hodnotou důvěry je považováno za podlaží, ve kterém se aktuálně nachází výtahová kabina. V případě, že se výsledek shoduje s podlažím cílovým, je program pozastaven. V opačném případě se pokračuje v jízdě a seznam hodnot důvěry je uložen jako vstup pro fúzi v dalším kroku.

6.3 Vizualizace výstupů

Na závěr jsou ukázány výsledky z aplikace. Prvotní ověření funkcionality proběhlo na výtahu v budově A3 na Fakultě strojního inženýrství v Brně. Jedná se o výtah, jehož minimální dosažitelné podlaží je číslo 1 a maximální dosažitelné podlaží je číslo 7. Maximálně je možné uskutečnit jízdu se změnou o 6 podlaží. V průběhu jízdy byly zaznamenány skutečná podlaží, kde kabina výtahu zastavila, aby bylo možné zpětně ověřit výsledek. Systém byl kalibrován při jízdě mezi podlažím 2 a 3. Poté byla prováděna detekce. Kabina výtahu zastavila během jízdy v podlaží 3, 6, 2 a 7. Výsledek detekce je zobrazen na Obr. 6.2.

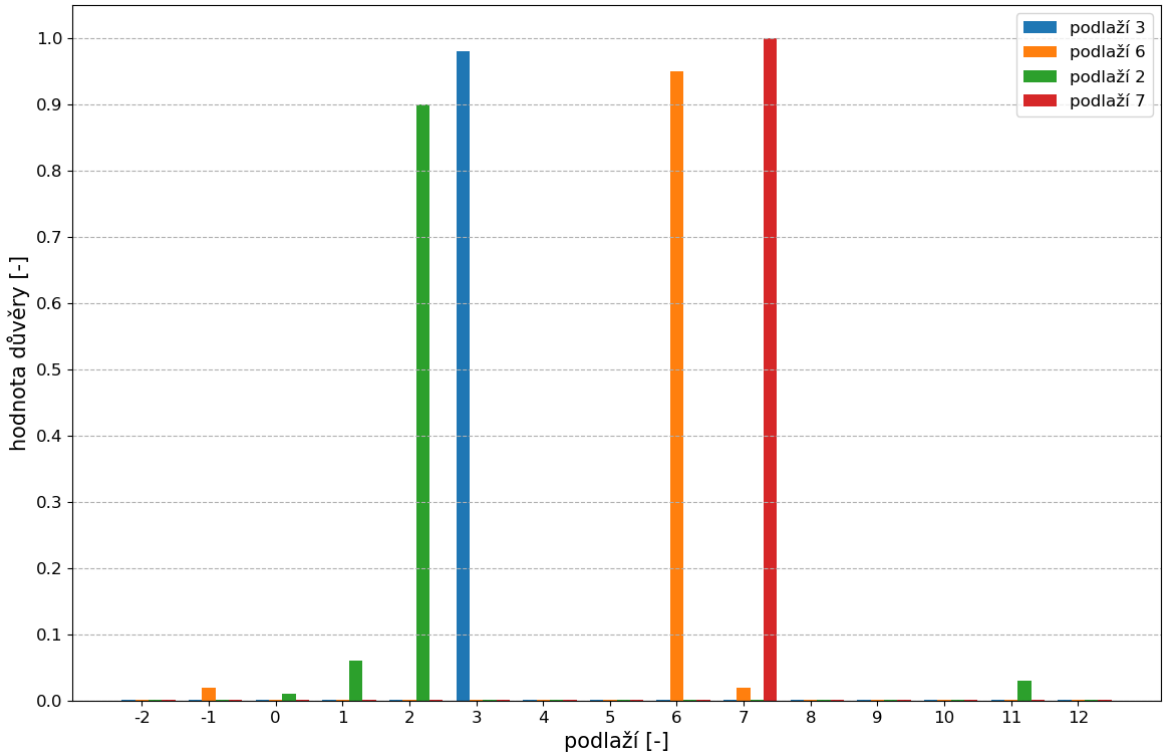


Obr. 6.2 – Výsledek detekce pater při jízdě výtahem

V první grafu odshora jsou zobrazeny výsledky jednotlivých částí programu v porovnání se skutečností. Jelikož je vyhodnocení prováděno pouze v okamžiku, kdy je detekováno zastavení

výtahu, jsou tato data diskrétní. Z výsledků je patrné, že skutečná patra odpovídají výstupům z datové fúze. Ve zbývajících grafech je pro názornější představu zobrazen průběh zrychlení a rychlosti v průběhu jízdy. Je možné vidět, že prvotní odhad výsledků algoritmu využívající akcelerometrická data byl v případě špatného výsledku opraven výstupem z algoritmu využívající obrazová data.

Dále byly uloženy seznamy hodnot důvěry pro všechna zastavení. Vizualizaci hodnot důvěry je možné vidět na Obr. 6.3. Jedná se o vizualizaci důvěry systému v detekované podlaží. Každému zastavení odpovídají hodnoty s příslušnou barvou. Jedná se tak o rozdělení pravděpodobnosti pro 4 různá zastavení. Nejvyšší hodnota určuje detekovanému podlaží.



Obr. 6.3 – Vizualizace důvěry v detekované podlaží

Z výsledků je patrné, že pokud je výstup z obou programů shodný, je výsledná pravděpodobnost rovna 1. V případě, že je měření akcelerometru či kamery odlišné, je výsledná hodnota důvěry nižší a část důvěry je přesunuta na podlaží, které odpovídá výstupu akcelerometru nebo kamery. Tímto testem bylo ověřeno, že systém se chová dle předpokladů.

7 VYHODNOCENÍ

7.1 Experimentální ověření

Celková úspěšnost systému byla vyhodnocena na pěti různých výtahových kabinách. Popis jednotlivých výtahů a ukázky displejů jsou zobrazeny v následující tabulce Tab. 7.1.

Tab. 7.1 – Popis a ukázka displeje výtahů pro ověření aplikace

Umístění: Fakulta strojního inženýrství, budova A3 Minimální podlaží: 1 Maximální podlaží: 7 Výrobce výtahu: OTIS	
Umístění: Fakulta strojního inženýrství, budova A4 Minimální podlaží: 1 Maximální podlaží: 7 Výrobce výtahu: OTIS	
Umístění: Halasovo náměstí, Lesná, Brno Minimální podlaží: 0 Maximální podlaží: 13 Výrobce: KONE	
Umístění: Příční, Zábrdovice, Brno Minimální podlaží: 0 Maximální podlaží: 5 Výrobce: KONE	
Umístění: Štefánikova čtvrť, Černá pole, Brno Minimální podlaží: 0 Maximální podlaží: 10 Výrobce výtahu: KONE	

Vyhodnocována byla synchronizovaná data z akcelerometru a z kamery. U dat z kamery bylo zaručeno, že je snímán pouze informační displej. Nejprve byla provedena kalibrace při jízdě o 1 podlaží. Poté byla provedena jízda výtahem. Z důvodu testování nebylo nastaveno finální podlaží, proto vyhodnocování probíhalo po celou dobu jízdy.

7.2 Výsledky

Výsledky byly uloženy a na základě nich bylo provedeno vyhodnocení.

Tab. 7.2 – Výsledky testování funkčnosti programu

Výtah	Počet správných detekcí	Počet chybných detekcí	Celkem
A3	24	0	24
A4	21	0	21
Lesná	26	1	27
Zábrdovice	11	1	12
Černá pole	34	1	35

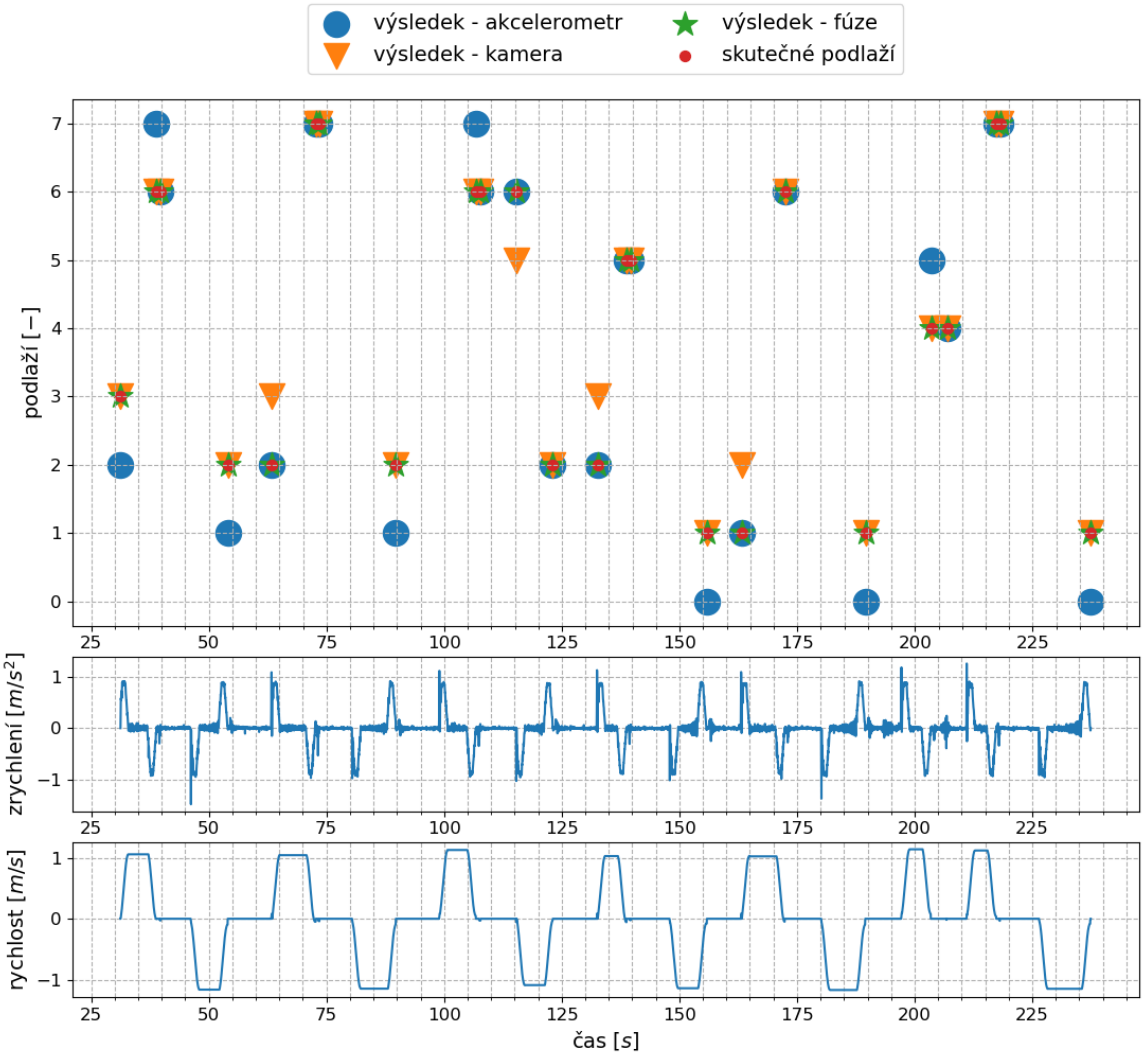
Na základě těchto výsledků byla vypočtena celková úspěšnost algoritmu na 97%.

V tabulce Tab. 7.3 je zobrazena úspešnost jednotlivých částí programu pro všechna měření.

Tab. 7.3 – Výsledky jednotlivých částí programu

	Počet chybných detekcí [-]	Procentuální úspěšnost [%]
Algoritmus obrazových dat	10	91
Algoritmus akcelerometrických dat	21	81
Datová fúze	3	97

Vizualizace výsledků testování ve výtahu v budově A3 na FSI je zobrazena na Obr. 7.1. Výsledky ostatních měření jsou uloženy v příloze ve složce *Results*.



Obr. 7.1 - Vizualizace výsledků jízdy výtahem v budově A3 na FSI

V grafu zobrazujícím závislost podlaží na čase je možné vidět pozorované výsledky jednotlivých částí v průběhu jízdy. Výsledky datové fúze odpovídají skutečnému průběhu jízdy. Z výsledků je patrné, že chybovost algoritmu závisejícím na akcelerometrických datech je vyšší. V 9 případech byl výsledek určen chybně. Tato chyba byla vždy taková, že bylo určeno podlaží o 1 patro vyšší nebo o 1 patro nižší. V případě chybného výsledku kamery se jedná především o chybné určení zastavení výtahu, kdy se na displeji objevuje číslo následujícího podlaží a výtah je v pohybu. Tento jev je způsobený především průběhem zrychlení konkrétního výtahu. Realita je však taková, že se výtahová kabina rozjíždí a dosahuje konstantního zrychlení. Výstupem algoritmu využívajícího akcelerometrická data je posunutí o 0 podlaží. Z toho důvodu je dána na 0 posunutí vysoká váha. Výsledkem datové fúze je, že se detekované podlaží nijak nemění a systém čeká na správnou detekci zastavení výtahu. Jev chybné detekce zastavení patra se v aplikaci objevuje, avšak v případě zapojení datové fúze nemá na výsledek žádný vliv. Tímto výsledkem je zároveň možné demonstrovat robustnost navrženého řešení. Ošetřit tuto chybu je také možné přímo v algoritmu, kdy nebude možné detekovat zastavení výtahu při změně patra nižší než 1. V aktuálním případě byl zvolen přístup datové fúze a toto řešení je dle prezentovaných výsledků správné.

8 ZÁVĚR

Cílem práce bylo vyvinout a implementovat metodu detekce aktuálního podlaží při průjezdu výtahem pomocí fúze dostupných senzorických dat. Na základě rešeršní části byly jednotlivé cíle blíže specifikovány a byl navržen systém, který odpovídá požadavkům. Bylo využito obrazových dat z kamery a akcelerometrických dat z inerciální měřící jednotky. Nejprve byly realizovány jednotlivé části samostatně, aby bylo možné ověřit navržená řešení a získat požadovaný vstup do datové fúze.

V první fázi byl vyvinut algoritmus využívající akcelerometrická data, který umožňuje ze zaznamenaného zrychlení vypočítat rychlost a polohu v reálném čase bez degradace dat driftem. Systém byl otestován na 16 různých výtahových kabinách. Na základě výsledků byla vytvořena tabulka podmíněných pravděpodobností, která určuje pravděpodobnost měřené změny podlaží za předpokladu skutečné změny podlaží. Se zvyšující se změnou podlaží pravděpodobnost správného výsledku klesá.

V druhé fází byl vyvinut systém využívající obrazová data. Displej výtahu poskytuje informaci o čísle aktuálního podlaží. Na základě klasifikace obrazu do klasifikačních tříd je možné určit aktuální podlaží. Nejprve byla získána obrazová množina, která zahrnuje obrázky displejů z více jak 20 výtahových kabin. Na základě těchto dat byla vybrána a natrénována CNN EfficientNet-B0. Tato síť provádí klasifikaci obrazového vstupu displeje výtahu. Výstupem je rozložení pravděpodobnosti klasifikačních tříd. Třída s nejvyšší hodnotou pravděpodobnosti určuje číslo aktuálního podlaží zobrazené na displeji a tím i číslo aktuálního podlaží. Klasifikace byla otestována na testovací datové sadě. Výsledná přesnost klasifikace byla určena na 96%. Výsledky byly zaznamenány do tabulky podmíněných pravděpodobností.

Závěrečnou fází bylo tyto informace využít a spojit v datové fúzi. Cílem fúze bylo zvýšit robustnost řešení. Na základě rešeršní části byl vybrán koncept Bayesova filtru. Tento přístup umožňuje využít pravděpodobnosti, které nemají normální rozdělení. Zároveň lze zahrnout pravděpodobnost výsledků z algoritmů na základě tabulek podmíněných pravděpodobností. Algoritmus byl úspěšně teoreticky ověřen.

Jednotlivé části byly spojeny ve výsledné aplikaci. Byla naprogramována aplikace, která umožňuje využít zaznamenaná data nebo data ze senzorů v reálném čase. Výsledný algoritmus dokáže detekovat zastavení výtahu a na základě této informace provede výpočet změny podlaží a klasifikaci aktuálního snímku displeje výtahu. Výsledky jsou dále spojeny v datové fúzi, jejíž výstupem je aktuální podlaží a příslušná velikost pravděpodobnosti, která určuje důvěru systému v detekované podlaží. Celý systém byl otestován na synchronizovaných datech z 5 různých výtahových kabin. Výsledná pravděpodobnost byla stanovena na 97%.

S ohledem na prezentované výsledky je možné považovat všechny cíle práce za splněné.

8.1 Budoucí směřování práce

Mezi hlavní nedostatky tohoto řešení patří nutnost kalibrace polohy před každou novou jízdou výtahem. Dále vysoká závislost na obrazových datech při vyšší velikosti změny podlaží, kdy akcelerometr nedosahuje vysokých přesností. S ohledem na tyto nedostatky je možné při dalším vývoji sytému zohlednit široké možnosti strojového učení. Díky množství akcelerometrických dat získaných v rámci práce by bylo možné aplikovat na data algoritmy, které by se na základě těchto dat dokázaly adaptovat na daný výtah. Jednalo by se např. o LSTM sítě. Jedná se RNN,

které jsou využívány při detekci sekvencí a časově závislých dat. Výsledkem by měla být vyšší přesnost a univerzálnost řešení.

Dalším omezením je maximální počet pater, které algoritmus dokáže detekovat. V případě požadavku jízdy ve vyšší budově je nutné rozšířit klasifikaci o nová obrazová data, provést vyhodnocení úspěšnosti a rozšířit tabulku podmíněných pravděpodobností.

Finálním krokem bude jednotlivé dílčí úlohy zmíněné v úvodu práce integrovat do fungujícího celku, který umožní autonomnímu systému úspěšně využívat přepravu výtahem.

Bibliografie

[1] SIMMONS, R., D. GOLDBERG, A. GOODE et al. Grace: An autonomous robot for the AAAI Robot Challenge. *AI Magazine* [online]. 2003, **24**(2), 51-72 [cit. 2020-11-19]. ISSN 07384602.

[2] WANG, W., Y. CHIEN, H. CHIANG, W. WANG a C. HSU. Autonomous Cross-Floor Navigation System for a ROS-Based Modular Service Robot. In: *2019 International Conference on Machine Learning and Cybernetics (ICMLC)*. 2019, s. 1-6. ISSN 2160-1348. Dostupné z: doi:10.1109/ICMLC48188.2019.8949176

[3] ABDULLA, Ali, Hui LIU, Norbert STOLL a Kerstin THUROW. A robust method for elevator operation in semi-outdoor environment for mobile robot transportation system in life science laboratories. In: *2016 IEEE 20th Jubilee International Conference on Intelligent Engineering Systems (INES)* [online]. IEEE, 2016, s. 45-50 [cit. 2020-11-30]. Dostupné z: doi:10.1109/INES.2016.7555147

[4] ISLAM, K.T., G. MUJTABA, R.G. RAJ a H.F. NWEKE. Elevator button and floor number recognition through hybrid image classification approach for navigation of service robot in buildings. In: *2017 International Conference on Engineering Technology and Technopreneurship, ICE2T 2017* [online]. Institute of Electrical and Electronics Engineers Inc, 2017, , s. 1-4 [cit. 2020-12-04]. ISBN 9781538618073. Dostupné z: doi:10.1109/ICE2T.2017.8215992

[5] LI, You, Zhouzheng GAO, Zhe HE, Peng ZHANG, Ruizhi CHEN a Naser EL-SHEIMY. Multi-Sensor Multi-Floor 3D Localization With Robust Floor Detection. *IEEE Access* [online]. 2018, **6**, 76689-76699 [cit. 2020-10-15]. ISSN 2169-3536. Dostupné z: doi:10.1109/ACCESS.2018.2883869

[6] KIM, Eun-Ho, Sang-Hyeon BAE a Tae-Young KUC. Mobile service robot multi-floor navigation using visual detection and recognition of elevator features(ICCAS 2020). *2020 20th International Conference on Control, Automation and Systems (ICCAS)* [online]. IEEE, 2020, , 982-985 [cit. 2020-12-

04]. ISBN 978-89-93215-20-5. Dostupné z: doi:10.23919/ICCAS50221.2020.9268202

[7] GAJDA, Janusz, Ryszard SROKA a Piotr BURNOS. Sensor Data Fusion in Multi-Sensor Weigh-In-Motion Systems. *Sensors* [online]. Basel: MDPI AG, 2020, **20**(12), 3357 [cit. 2020-11-19]. Dostupné z: doi:10.3390/s20123357

[8] HALL, David L. (David Lee). *Mathematical techniques in multisensor data fusion*. 1. Boston: Artech House, 1992, xi, 301 s. : il. ; 24 cm. ISBN 0-89006-558-6.

[9] THRUN, Sebastian, Wolfram BURGARD a Dieter FOX. *Probabilistic robotics*. 1. Massachusetts: MIT Press, 2006, xx, 647 s. : il. ISBN 0-262-20162-3.

[10] VĚCHET, Stanislav, Vít ONDROUŠEK a Jiří KREJSA. Sensors data fusion via Bayesian filter. *Proceedings of 14th International Power Electronics and Motion Control Conference EPE-PEMC 2010* [online]. IEEE, 2010, , - [cit. 2021-03-22]. ISBN 978-1-4244-7856-9. Dostupné z: doi:10.1109/EPEPEMC.2010.5606874

[11] GREWAL, Mohinder a Angus ANDREWS. *Kalman filtering: theory and practice using MATLAB*. 3rd ed. Hoboken: Wiley, 2008, xvi, 575 s. : + 1 CD-ROM. ISBN 978-0-470-17366-4.

[12] ULUSOY, Melda. Understanding Kalman Filters: Part 4: An Optimal State Estimator Algorithm. In: *Mathworks.com* [online]. [cit. 2021-02-14]. Dostupné z: https://www.mathworks.com/videos/understanding-kalman-filters-part-4-optimal-state-estimator-algorithm--1493129749201.html

[13] ZHAO, He a Zheyao WANG. Motion Measurement Using Inertial Sensors, Ultrasonic Sensors, and Magnetometers With Extended Kalman Filter for Data Fusion. *IEEE Sensors Journal* [online]. IEEE, 2012, **12**(5), 943-953 [cit. 2021-03-08]. ISSN 1530-437X. Dostupné z: doi:10.1109/JSEN.2011.2166066

[14] QIU, Jun-wei a Yu-chee TSENG. M2M Encountering: Collaborative Localization via Instant Inter-Particle Filter Data Fusion. *IEEE Sensors Journal* [online]. IEEE, 2016, **16**(14), 5715-5724 [cit. 2021-03-06]. ISSN 1530-437X. Dostupné z: doi:10.1109/JSEN.2016.2566679

[15] O' MAHONY, Niall, Sean CAMPBELL, Carvalho ANDERSON, Suman HARAPANAHALLI, Gustavo VELASCO-HERNANDEZ, Lenka KRPALKOVA, Daniel RIORDAN a Joseph WALSH. Deep Learning vs. Traditional Computer Vision. *ArXiv.org* [online]. Ithaca: Cornell University Library, arXiv.org, 2019 [cit. 2021-03-11]. Dostupné z: doi:10.1007/978-3-030-17795-9

[16] ARAUJO DOS SANTOS, Leonardo. Neural Networks. *Artificial intelligence* [online]. [cit. 2021-03-11]. Dostupné z: https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/supervised_learning/neural_networks

[17] SUMIT, Saha. A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way. *Towards Data Science* [online]. [cit. 2021-03-11]. Dostupné z: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[18] ČERMÁK, Marek. *Metody využívané pro OCR*. Vysoké učení technické v Brně. Fakulta strojního inženýrství, 2018, 76 s. Bakalářská práce. Vedoucí práce Ing. Daniel Zuth, Ph.D.

[19] MOHAJON, Joydwip. Confusion Matrix for Your Multi-Class Machine Learning Model. *Towards Data Science* [online]. [cit. 2021-04-27]. Dostupné z: https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826

[20] TAN, Mingxing a Quoc LE. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ArXiv.org* [online]. Ithaca: Cornell University Library, arXiv.org, 2020 [cit. 2021-03-30]. Dostupné z: http://search.proquest.com/docview/2231642984/

[21] ČERNIL, Martin. *Automatická detekce ovládacích prvků výtahu zpracováním digitálního obrazu.* [online]. Brno, 2021 [cit. 2021-05-04]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/132984.. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav mechaniky těles, mechatroniky a biomechaniky. Vedoucí práce Jiří Krejsa.

[22] SEKACHEV, Boris, Nikita MANOVICH a Andrey ZHAVORONKOV. Computer Vision Annotation Tool: A Universal Approach to Data Annotation. *Intel* [online]. [cit. 2021-04-27]. Dostupné z: https://software.intel.com/content/www/us/en/develop/articles/computer-vision-annotation-tool-a-universal-approach-to-data-annotation.html

[23] MPU-9250 Product Specification: Revision 1.1. *InvenSense* [online]. [cit. 2021-04-29]. Dostupné z: https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf

[24] TAYLOR, Brian. MPU 9250 library. In: *Github* [online]. [cit. 2021-04-27]. Dostupné z: https://github.com/bolderflight/mpu9250

[25] RASHINKAR, Pratiksha a V. KRUSHNASAMY. An overview of data fusion techniques. In: *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)* [online]. Bangalore, India: IEEE, 2017, , s. 694-697 [cit. 2020-10-06]. ISBN 978-1-5090-5960-7. Dostupné z: doi:10.1109/ICIMIA.2017.7975553

[26] JAROSZEK, P. a M. TROJNACKI. Localization of the wheeled mobile robot based on multi-sensor data fusion. *Journal of Automation, Mobile Robotics and Intelligent Systems* [online]. Industrial Research Institute for Automation and Measurements, 2015, **9**(3), 73-84 [cit. 2020-12-02]. ISSN 18978649. Dostupné z: doi:10.14313/JAMRIS_3-2015/26

[27] GAO, Bingbing, Gaoge HU, Shesheng GAO, Yongmin ZHONG a Chengfan GU. Multi-Sensor Optimal Data Fusion Based on the Adaptive Fading Unscented Kalman Filter. *Sensors* [online]. Basel: MDPI AG, 2018, **18**(2), 488 [cit. 2021-02-14]. Dostupné z: doi:10.3390/s18020488

[28] VĚCHET, S. a J. KREJSA. Sensors Data Fusion via Bayesian Network. *Recent Advances in Mechatronics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, , 221-226. ISBN 978-3-642-05021-3. Dostupné z: doi:10.1007/978-3-642-05022-0_38

[29] SINGHAL, A. a C. BROWN. Dynamic bayes net approach to multimodal sensor fusion. In: *Proceedings of SPIE - The International Society for Optical*

Engineering [online]. 1997, , s. 2-10 [cit. 2021-03-09]. ISBN 9780819426413. ISSN 0277786X. Dostupné z: doi:10.1117/12.287628

Seznam obrázků

Seznam tabulek

Seznam zkratek

ANN – Artificial Neural Network
CNN – Convolution Neural Network
ROS – Robot Operating System
OCR – Optical Character Recognition
ROI – Region of Interest
EKF – Extended Kalman Filter
KF – Kalman Filter
RSS – Receive Signal Strength
CV – Computer Vision
IMU – Inertial Measurement Unit
DLPF – Digital Low Pass Filter

Kapitola 8

Seznam příloh

Z důvodu vysokého datového objemu příloh budou veškeré přílohy přístupné u vedoucího práce.

1. src
    1.1. core
        1.1.1. accelerometer_estimation
            1.1.1.1. acc_estimate_floor.py
        1.1.2. camera_estimation
            1.1.2.1. camera_estimate_floor.py
            1.1.2.2. data_preprocessing.py
            1.1.2.3. nn_model.py
        1.1.3. bayes_filter
            1.1.3.1. bayesFilter.py
        1.1.4. utils
            1.1.4.1. utilsClass.py
            1.1.4.2. complex_visualization.py
    1.2. input_feed
        1.2.1. acc_feed
            1.2.1.1. acc_realtime_feed.py
            1.2.1.2. csv_file_feed.py
            1.2.1.3. AccFeed.py
        1.2.2. image_feed
            1.2.2.1. camera_feed.py
            1.2.2.2. video_feed.py
            1.2.2.3. ImageFeed.py
        1.2.3. InputFeed.py
    1.3. main
        1.3.1. flags_global.py
    1.4. App.py
    1.5. main.py
2. MPU9250_library
3. notebooks
    3.1. display_digit_classification_learning.ipynb
4. data
    4.1. accelerometer_data
    4.2. fusion_data
    4.3. machine_learning_data
5. Results

Kapitola 8

5.1. floor_detection_results
5.2. training_checkpoints

Tabulka A

	P(změna|realná změna)														
reálná změna patra	-	-14	-13	-12.0	-11.0	-10.0	-9.0	-8.0	-7.0	-6.0	-5.0	-4.0	-3.0	-2.0	-1.0	0
	-14	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	-13	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-12	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	-11	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0
	-10	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0
	-9	0	0	0	0	5	6	1	0	0	0	0	0	0	0	0
	-8	0	0	0	0	1	3	12	0	0	0	0	0	0	0	0
	-7	0	0	0	0	0	0	3	7	0	0	0	0	0	0	0
	-6	0	0	0	0	0	0	0	5	15	1	0	0	0	0	0
	-5	0	0	0	0	0	0	0	0	8	15	1	0	0	0	0
	-4	0	0	0	0	0	0	0	0	0	4	26	1	0	0	0
	-3	0	0	0	0	0	0	0	0	0	0	1	17	0	0	0
	-2	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0
	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	21	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30

	P(změna|realná změna)															
reálná změna patra	-	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	37	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0
	3	0	0	0	16	2	0	0	0	0	0	0	0	0	0	0
	4	0	0	0	1	19	3	0	0	0	0	0	0	0	0	0
	5	0	0	0	0	1	18	4	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	1	14	4	0	0	0	0	0	0	0
	7	0	0	0	0	0	0	1	9	4	0	0	0	0	0	0
	8	0	0	0	0	0	0	0	0	12	5	0	0	0	0	0
	9	0	0	0	0	0	0	0	0	2	4	5	0	0	0	0
	10	0	0	0	0	0	0	0	0	0	1	3	1	0	0	0
	11	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0
	12	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
	13	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	14	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-